DISJOINT TALLY METHOD: A MONTE CARLO SCORING METHOD USING COMPRESSED SENSING TO REDUCE STATISTICAL NOISE AND MEMORY

A Dissertation

by

JONATHAN ROBERT MADSEN

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Ryan G. McClarren
Co-Chair of Committee,	John Ford
Committee Members,	Simon Foucart
	Sunil S. Chirayath
Head of Department,	Yassin A. Hassan

December 2017

Major Subject: Nuclear Engineering

ABSTRACT

We present a methodology for using compressed sensing concepts to compute a global solution for Monte Carlo simulations with less memory and an accelerated reduction in statistical noise. The methodology utilizes a process known as total variation minimization to reconstruct the solution from the projection of the scoring array onto randomly constructed basis sets.

In the disjoint tally method, the memory allocation of the basis sets and the projection is a fraction of the size of a standard scoring array and is a replacement for the array at runtime. Memory reduction results are approximately 25% for 1 thread and 1 tally quantity, ~35% for 16 threads and 1 tally quantity, and can exceed 75% at 16 threads and 8 tally quantities.

The reconstruction method is peak-preserving and applies statistical denoising upon reconstruction. If a satisfactory global solution to scoring array A can be reached at f_N, where N is the number of particles simulated and $i \in 1, \ldots, N$, then if Δ_i is the difference between f_i and f_N, our method at f_i consistently produces a smaller Δ_i than a standard scoring array at f_i. Since N is arbitrary, the disjoint tally method effectively decreases computation time by producing results at f_p where $i < p < N$.

We begin by introducing the concept of disjoint tallies and provide the procedure for local reconstruction on subsets of the global mesh. We present evidence of the validity of the solution produced by the methodology and the reduction in memory footprint via direct comparison to a memory-efficient storage implementation within the Monte Carlo transport toolkit, Geant4. Additionally, we present a method for reconstructing statistical

quantities in the form of the variance, relative error, coefficient of variation, and root-mean-squared. Results are given for three different global reconstruction scenarios: a reactor bundle, a neutron shielding problem, and set of CT scans rendered directly from DICOM files.

We believe the demonstration of the significant reduction of data allocation size, the evidence of acceleration towards the bounded total variation of the solution, ability to reconstruct the quantities required for statistical checks, and the local nature of our reconstruction will provide capabilities necessary for high-fidelity exascale computing.

DEDICATION

For my late grandfather, Granddaddy John.

He received two Silver Snoopy Awards (Astronaut's Personal Achievement Award for Professional Excellence), the second of which was in recognition of his being a NASA Space Pioneer in the field of engineering analysis, design, and development of advanced aircraft and spacecraft systems. He also received the Presidential Medal of Freedom Award to the Apollo XIII Mission Operations Team among others awards for the Apollo missions. When I was a kid, he taught me about computers, how to shoot a rifle, and how to solder a circuit board along with so many other things. He was the smartest person I've ever known and the reason I wanted to pursue engineering ever since I was a boy. If he were still alive today, he'd probably be in the contributors section from us walking through every line of code together. I wish he could have been here for this.

ACKNOWLEDGMENTS

I would like to thank my parents most of all, who always provided me with the best opportunities in life.

I would like to thank Dr. Ryan McClarren, not only for his help and guidance throughout my graduate studies but for providing me with my first research job as an undergraduate student, I never would have found my love for research without it.

I would also like to thank Dr. Gamal Akabani, for taking a chance on an undergraduate with abysmal grades and getting him into graduate school — I truly don't know where I would be today if it weren't for your faith in my abilities.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Professor Ryan G. McClarren, Professor John Ford, and Professor Sunil S. Chirayath of the Department of Nuclear Engineering and Professor Simon Foucart of the Department of Mathematics.

The core for the reconstruction library in Chapter 3 was built from the ℓ_1-magic software package. [1]

All other work conducted for the dissertation was completed by the student independently.

Funding Sources

Graduate study was supported by the Center for Exascale Radiation Transport (CERT) at Texas A&M University through a research grant from the National Nuclear Security Agency (NNSA) under the Predictive Science Academic Alliance Program (PSAAP-II).

NOMENCLATURE

MC	Monte Carlo
CS	Compressed Sensing
DICOM	Digital Imaging and Communications in Medicine (Standardized format for medical imaging)
PDF	Probability Density Function
CDF	Cumulative Distribution Function
Step	Geant4 term for smallest unit of simulation for a particle. It may be a process the particle undergoes or a transport step
Track	Geant4 term for collection of steps spanning the creation of the particle until termination at zero kinetic energy, absorption, or exiting "world" volume
Event	Geant4 term for one or more starting particle(s)
Run	Geant4 term for accumulation of all of the events
Tally	Monte Carlo term for a quantity such as energy deposit that is recorded during the simulation
Scorer	Synonym for tally used by Geant4
Thread	Smallest sequenced of programmed instructions that can be managed independently by a scheduler, which is typically part of an operating system
Multithreading	Designating portions of an application to execute independently and concurrently (limited by # of CPU cores) on separate threads

MFP	Mean free path. The average distance a particle will travel before interacting

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

TABLE Page

1. INTRODUCTION

1.1 Overview

This dissertation presents a signal processing technique (compressed sensing) applied to a highly-accurate but computationally-expensive statistical method used for nuclear physics (Monte Carlo). The region of application is the scoring (also called "tally") system of quadrilateral or hexahedral meshes. When utilized within the framework of the Monte Carlo code, this methodology consumes significantly less memory and provides denoising to the statistical results inherent to Monte Carlo calculations. Additionally, the methodology can be applied after the simulation to standard tally results for the statistical denoising property.

Sections 1.2, 1.3, and 1.5 have been provided as an overview of the Monte Carlo method in nuclear physics and its computational challenges. Those familiar with the Monte Carlo method for nuclear physics may skip to Section 1.6.

In Section 2.1, an example of compressed sensing is given to provide a reference point to how memory compression is achieved and Sections 2.2, 2.3, and 2.4 provide the background on the mechanisms through which statistical denoising is accomplished.

In Sections 3.1 and 4, the formulation of the scoring system in a Monte Carlo code is provided in addition to details of our software development. Section 3.3 provides a proof-of-concept and demonstrates the ability to apply the statistical denoising on existing results.

Sections 3.4 and 3.5 detail the three different problem types and the scoring quantities presented in our results followed by a discussion about variance and conservation in Section 4.5 and 4.6, respectively.

Section 5.2 presents evidence of the peak-preservation in the statistical denoising. Section 5.3 uses the method for evaluating the predictive power of the statistical denoising outlined in Section 5.1. Memory reduction results are presented in Section 5.5 followed by a discussion of the reconstruction parameters and a summary of the reconstruction results in Sections 4.3 and 5.6.

In Appendix A, details about the Geant4 physics lists and simulation parameters can be found for the three geometries. Appendix B contains some supplementary results to Section 5.3 and reconstructions of statistical quantities supplementing 5.4. Appendix C contains numerous comparisons of all the Geant4 tallies with variations in a parameter called the "subrate," which is introduced in Section 3.1.

1.2 Monte Carlo Method in Nuclear Physics

Subatomic particles have many different interaction pathways and the prediction of which pathway the particle will take at a given position, time, and energy is non-deterministic. This stochastic nature of particle interactions creates a complex multivariate integral and Monte Carlo integration is a widely utilized calculation method to determine the overall behavior of the system.

The Monte Carlo technique uses a random number generator to randomly select from a probability density function (PDF) of interactions. In nuclear physics, these interaction probabilities are called "cross-sections," which are given in units of area, and denoted σ_x where x is the interaction type. When multiplied by the atomic density of the material,

this probability is transformed into units of inverse length and denoted Σ_x. This inverse is the average distance traveled between interactions of type x. In very general terms, the cross-sections are dependent on the type of particle being transported, the energy of the particle, and the composition and density of the material the particle is being transported through.

In a Monte Carlo simulation for nuclear physics, a particle is "created" at a certain point in space and given an energy and momentum direction. A series of random samples are taken from the PDF, and each sample defines an interaction from which the particle will be tracked in time and space in a series of "steps". Each particle step is evaluated independently based on the PDF of the particle state at the time of the step. Each step will determine the future state of the particle, which includes:

- Distance in space the particle travels
- Any energy change in the particle
- Any secondary particles[1]
- Any change in momentum direction
- Any change in particle charge

In some instances, the particle will lose only a fraction of its energy and the sampling procedure will repeat itself but be re-evaluated at the new particle state or, in other instances, the particle may lose all of its kinetic energy[2] and the tracking of the particle will be stopped.

[1] *e.g.,* For example, the particle may cause an electron from an atom in the material to be ejected

[2] Monte Carlo codes have customizable energy cut-off levels below which the particle is considered to have zero kinetic energy and tracking is stopped.

By repeating the above procedure for many, many particles, the macroscopic effect(s) of these individual interaction probabilities can be quantified. For example, several million x-rays can be transported in a Monte Carlo simulation to determine the distribution of energy deposited to a person in a CT scan or several hundred batches of several thousand neutrons can be simulated to calculate whether the reactor can sustain a chain reaction.

1.3 Issues with Monte Carlo Method

Computation of a quantity in Monte Carlo requires a large number of particles to be simulated. The standard deviation of the estimated value is proportional to $\frac{1}{\sqrt{N}}$ where N is the number of estimates of the value. In order to get a statistically significant result, the number of particles must be large for each individual volume of interest. This requirement can be accelerated by various variance reduction methods, such as weight windows and geometric importance sampling, which are described in Section 1.5.

Particle tracking is expensive when the number of volumes is large and the solution is required everywhere with a high resolution (low standard deviation) due to boundary checking during transport and the number of particle tracks that must be simulated to reach all volumes. As the problem becomes larger or more finely discretized, the number of tallies for each volume increases the memory footprint. The increase in memory footprint is compounded by the common requirement in Monte Carlo codes to provide information on the reliability of the statistical data, which is detailed in Section 1.4.

Monte Carlo simulations are often parallelized using distributed memory parallelism (*e.g.,* MPI) by one of two methods: (1) particle-decomposition, which replicates the full geometry on each node and each node has a different initial random number seed or (2)

domain-decomposition, which divides the geometry between the nodes and particles passing between the domains are transferred to the node on which the domain resides.

Particle-decomposition in Monte Carlo simulations is commonly referred to as embarrassingly parallel, only different initial random number seeds are required for the processes to be run in parallel. However, due to the large duplication in memory used in particle-decomposition from distributed memory parallelism, some Monte Carlo simulations toolkits have opted to introduce shared memory parallelism (*i.e.,* multithreading) in order to reduce memory footprints. In shared memory parallelism, the static portions of the simulation (*e.g.,* geometry, cross-section tables) are shared among all threads and the dynamic portions of the simulation (*e.g.,* scoring tallies) are, for optimal performance, stored in thread-local memory. If the requirement for memory provided by multithreading with thread-local scoring is still too large, some simulations are capable of implementing thread-global scoring with the appropriate locking mechanisms around the dynamic portions to prevent the classic data-race scenario (one thread trying to read a memory location while another thread attempts to modify the memory location).

The methodology presented in this thesis, which can be implemented in either distributed or shared memory parallelism, provides an alternative method to reducing the memory footprint in the scoring tallies in large problems where the scoring volumes are continuous and spatially consistent along the axis *i.e.,* a quadrilateral or hexahedral mesh.

1.4 Variance in Monte Carlo

Let us consider n independent random observations, μ_1, μ_2, ..., μ_n of a random variable. If this random variable ϕ_i is a function of t with a PDF $f(t)$, the expected value of ϕ is defined by

$$E(\phi) = \int \phi(t)\, f(t)\, dt\,, \tag{1.1}$$

where t is a variable representing all possible random paths. The variance of Eqn 1.1 is defined by

$$\text{Var}(\phi) = E(\phi^2) - \left[E(\phi)\right]^2. \tag{1.2}$$

The estimate of the expected value by the average of n samples is defined by

$$E(\phi) \simeq \bar{\mu_n} = \frac{1}{n}\sum_{i=1}^{n} \mu_i\,, \tag{1.3}$$

and according to the law of large numbers, the average $\bar{\mu_n}$ approaches the expected value $E(\phi)$ with a probability that approaches 1 as the sample size goes to infinity.

The Central Limit Theorem gives precise information on the convergency of the estimation, but requires the knowledge of the variance. Thus, the calculation of the variance is an important component of Monte Carlo simulations as it allows the determination of the confidence interval, which is defined as a range of values that there is a specified probability that the value of a parameter lies within it.

In order to estimate the variance without storing every observation, three quantities must be recorded:

- the sum of each estimate of the recorded quantity $\left(\sum_{i=1}^{N} x_i\right)$
- the sum of each estimate of the recorded quantity squared $\left(\sum_{i=1}^{N} x_i^2\right)$
- the number of estimates (N)

and the variance is calculated by

$$\sigma^2 = \frac{1}{N-1}\left[\sum_{i=1}^{N} x_i^2 - \frac{\left(\sum_{i=1}^{N} x_i\right)^2}{N}\right] \tag{1.4}$$

There are many additional quantities that can be calculated to check the statistical convergence. The root mean square (RMS)

$$RMS \equiv \sqrt{\frac{N}{N-1} * \left(\frac{\sum_{i=1}^{N} x_i^2}{\sum_{i=1}^{N} x_i} - \left[\frac{\sum_{i=1}^{N} x_i}{\sum_{i=1}^{N} w_i}\right]^2\right)} \tag{1.5}$$

can serve as a measure of how far on average the the error is from zero when the sum of the weight of each estimate $\left(\sum_{i=1}^{N} w_i\right)$ is also recorded. The relative error

$$\epsilon \equiv \frac{1}{\sqrt{N}} * \frac{\sigma}{\bar{\mu}} \tag{1.6}$$

serves as a measurement of precision of the recorded value and the coefficient of

variation

$$c_v \equiv \frac{\sigma}{\bar{\mu}} \tag{1.7}$$

is useful as a measurement of the *relative* standard deviation.

1.5 Variance Reduction in Monte Carlo

As mentioned in Section 1.3, a large number of estimates are required to reduce the variance of a recorded quantity. In many computations, especially ones requiring a global solution, certain regions of interest receive significantly reduced particle flux.

For example, imagine a company is building a structure to house a neutron source. The company specifies that neutron source will be placed at the east end of the building and the team of scientists using the neutron source will be placed at the west end. A radiation protection group is tasked with determining the minimum distance the offices can be placed away from the neutron source to reduce construction costs and the ideal materials that should be used as a neutron shield. The material aspect would be an optimization application requiring several iterations of the same geometric configuration with a substitution of different materials. The distance determination requires a global solution for the geometry. Neutrons are extremely harmful to biological tissue so the determination of the flux of neutrons reaching the scientist's offices is required to a high degree of accuracy.

The radiation protection group chooses Monte Carlo transport as the simulation method for its accuracy but finds only approximately one in 1,000 neutrons reach the west end of the simulation. However, the group must determine if that one in 1,000 neutron multiplied by the source rate and integrated over fifty 40-hour work weeks results in the

workers exceeding the minimum acceptable yearly dose to the workers strictly enforced by the company. The radiation protection group finds the average simulation time of transporting one neutron is 0.5 seconds and around 100 estimates for each voxel in the global solution is required for sufficient precision. If the offices are discretely divided into 100,000 voxels and the group has access to a server with 1024 nodes, one analog[3] would require a minimum of 56.5 days to compute. The time required for each one of these calculations would make any optimization of the material infeasible.

With this example in mind, we introduce variance reduction methods in Monte Carlo. These methods are utilized to reduce the computation time spent in regions of low interest and increase computation time in regions of higher interest. These regions of high interest can be geometrical regions like the previous example but can also be energy regions. Ideally, each region would see the same distribution and flux of particles for maximum computational efficiency but source-driven problems create an imbalance where regions closer to the source receive a broader distribution and higher flux than regions further away. Variance reduction methods, when properly utilized, help correct this imbalance.

Two popular methods of variance reduction are used in this dissertation[4]:

1. Weight Windows

2. Importance Weighting

Both methods, in their own fashion, employ particle splitting and "Russian Roulette". The method of particle splitting involves taking a given particle and duplicating it. Each particle is then assigned a statistical weight corresponding to the number of duplicates that

[3] Analog mode is particle transport without variance reduction methods

[4] Other methods include implicit capture, forced collisions, and exponential transform [3]

were created and with respect to the parent particle weight. For example, if the original particle had a statistical weight of 1, then upon splitting, both the original particle and its duplicate would have a statistical weight of 0.5. If one of these particles are then split twice, the particle that is being split and its three duplicates would have a statistical weight of 0.125 $\left({}^{0.5}/_{4}\right)$. The Russian Roulette method is the reverse of particle splitting. If a particle is undergoing Russian Roulette, it has a probability of survival (commonly 0.5), and a random number is generated between 0 and 1 and if the random number is less than the probability of survival, the particle weight is increased by the inverse of the probability of survival (*e.g.,* doubled if the probability of survival is 0.5) but if the random number is greater than the probability of survival, the particle is killed. In the end, the sum of the statistical weights must equal 1 and all corresponding quantities that are accumulated with respect to the particle (*e.g.,* energy deposit, flux, track length, etc.) are multiplied by the statistical weight, although certain exceptions exists such as a tally for the minimum kinetic energy at generation.

These two common methods of variance reduction are commonly used together and can be employed in a number of fashions to statistically resolve spatial or energy deficient regions of the simulation. Importance weighting, in particular, is commonly used for shielding type problems where certain regions of the geometry receive a low number of particles in analog mode. In importance weighting, certain real or artificial boundaries are imposed that split particles heading towards regions of interest and employ Russian Roulette onto particles heading away from the regions of interest — thereby, effectively "pushing" more particles towards these regions of interest. A more detailed description of Monte Carlo variance reduction can be found in [3].

1.6 Compressed Sensing

Compressed sensing is a well-established methodology [1, 4–6] and has seen a significant increase in interest in recent years. Compressed sensing is a signal processing technique for efficiently acquiring and reconstructing a signal by finding solutions to under-determined linear systems. It is based on the principle that, through optimization, the sparsity of the signal can be exploited from fewer samples required by the Shannon-Nyquist sampling theorem.

The Shannon-Nyquist sampling theorem [7, 8] states that if the signal's highest frequency is less than half of the sampling rate, then the signal can be reconstructed perfectly. It is a fundamental bridge between continuous time signals and discrete time signals to establish a sufficient condition for a sample rate that permits a discrete sequence of samples to capture all the information from a continuous-time signal of finite bandwidth. In essence, the Shannon-Nyquist sampling theorem defines the minimum amount of data that must be acquired in order to recover the solution.

Compressed sensing has two conditions under which recovery is possible from far fewer samples required by the Shannon-Nyquist theorem:

1. Sparsity — the signal must be sparse in some basis

2. Incoherence

A basis is a set of vectors that are linearly independent and every vector in the vector space is a linear combination of this set. A signal that is sparse in some basis means that the signal has a small number of non-zero coefficients. For a 2-D signal, $\mathrm{e}_1 = (1, 0)$ and $\mathrm{e}_2 = (0, 1)$ are a basis and all entries in the signal would be a linear combination of

$\alpha_1 \mathrm{e}_1 + \alpha_2 \mathrm{e}_2$, where α_1 and α_2 are constants. For a 1-D signal, $\mathrm{e}_1 = (1)$ is a basis and all entries in the signal would be a linear combination of $\alpha_1 \mathrm{e}_1$. Since the signal is required to be sparse with respect to the basis, the large majority of entries composing the signal belong to $\mathbf{R}^{\mathrm{n}}$ would have $\alpha_1 = \ldots = \alpha_n = 0$.

With respect to Monte Carlo simulations in nuclear physics, it cannot guaranteed that the signal is sparse in some basis, nor is it highly probable. Therefore, the sparsity condition does not apply. However, two bases are said to be coherent when they have a large value when integrated against each other [9] and incoherence can be almost guaranteed with any basis where the sampling procedure is random [5]. These properties allow the incoherence condition to be the foundation of our methodology.

1.7 Compressed Sensing in Monte Carlo Problems

Compressive sensing can be applied to a particular subset of problem types for Monte Carlo simulations. The requirements are as follows:

- Scoring volumes within mesh must not overlap
- Scoring volumes within mesh must share boundaries
- The scoring mesh must be decomposable to a symmetric number of voxels in each dimension by an integer factor (*e.g.,* $R^{M \times N}$ where $N = \alpha * M$ or $M = \alpha * N$ and α is an positive integer)

The method of reconstruction (*e.g.,* ℓ_1 minimization or total variation minimization) depends on the problem type. For the majority of quantities in Monte Carlo simulations, reconstruction with total variation minimization is the appropriate reconstruction method.

In order to achieve a reduction in runtime memory requirements, the problem will ideally use multithreading and have a very large number of scoring volumes. The total variation minimization (TVM) reconstruction method produces highly accurate solutions (with respect to the true solution) when the true solution is smooth. When the true solution is very noisy, the TVM method will denoise the solution while preserving peaks and edges at a low *subrate* (to be discussed later) setting and, as the subrate setting is increased, the reconstructed solution will more faithfully represent the simulated solution. However, due to the statistical nature of Monte Carlo simulations, a decrease in the difference between the reconstruction and the standard tallies is not entirely desirable. We will present evidence in Section 5.1 that the total variation of tallies in Monte Carlo simulations reduces as the number of simulated particles increases. Therefore, the denoising of the solution with a lower subrate setting is a desirable property since simulating more particles is, effectively, a denoising process itself. In addition to the subrate parameter, there is another parameter, ϵ, that allows the user to modify how faithfully the simulated solution is represented — *i.e.,* a larger ϵ will produce more statistical denoising than a smaller value of ϵ.

1.8 Choice of Algorithms

In Section 2.1, we will present an example of an application of compressed sensing. The recognition of the capability to reinterpret the concepts introduced in a Monte Carlo transport simulation was the source of inspiration for the development of this methodology. There are many algorithmic variations in compressed sensing at many different levels. Our methodology does not claim that our choices are the ideal algorithmic choices. Our choices reflect the best-suited selection of algorithms based on the variations provided in the ℓ_1-magic software package [1] and future work will include

alternative choices that we have learned are generally considered superior for our dataset.

At the highest level, the first choice is the selection of quantity to be minimized. This choice can be our choice, the total variation, for reasons to be discussed later. Another popular choice can be the ℓ_1 norm and the selection of the quantity to be minimized depends on the properties of the dataset. Other algorithmic choices on difference levels include the interior point method (our choice was the log-barrier) and the constraints for the minimization.

In Sections 2.2 and 2.3, we discuss the total variation minimization and in Section 2.4, we discuss the log-barrier interior point method. The total variation, defined in Section 2.2, is a measurement of the total change in the local gradients in the dataset and we must address why the total variation in a Monte Carlo calculation is significant and why the total variation was selected as the ideal quantity to minimize in a Monte Carlo transport simulation.

The most significant issue in Monte Carlo transport calculation solutions is the statistical noise from the sampling of the PDFs discussed in Section 1.4. Beyond a certain point, where enough particles have been simulated to reach all the regions of the problem (which we will call $t_{\min}$), all the ensuing (and significant) computational effort is in pursuit of reducing the statistical noise to ensure the quantities reported are statistically significant. Without the assurance of statistical significance, the solution is under-sampled and therefore, potentially invalid. A Monte Carlo transport solution run for infinite amount of time would result in zero statistical noise but a finite total variation, except in the trivial case of a vacuum. Since statistical noise increases the total variation, the minimization of the total variation can be interpreted as the goal for a Monte Carlo

transport simulation after $t_{\min}$ and is therefore our chosen minimization quantity.

2. PREVIOUS WORK

2.1 Applications Example — Single Pixel Camera

The amount of previous work in compressed sensing applications is extensive. Applications of compressed sensing range from a single pixel camera (SPC) [2], magnetic resonance imaging, and radar [4]. An excellent example of the application of compressed sensing can be seen in the Single Pixel Camera (see Fig. 2.1).

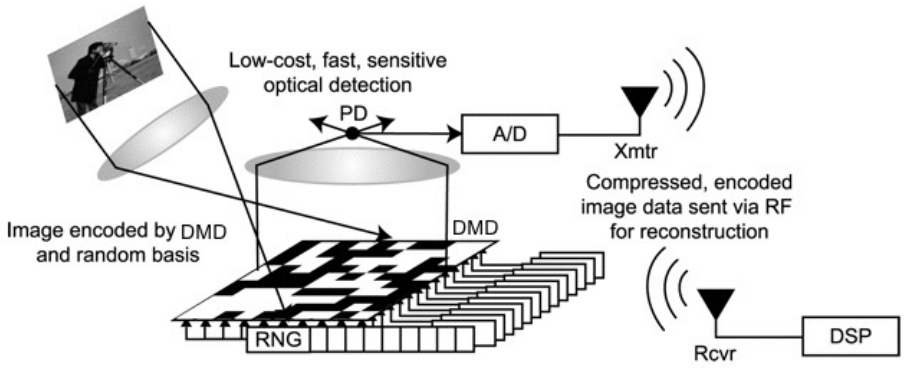
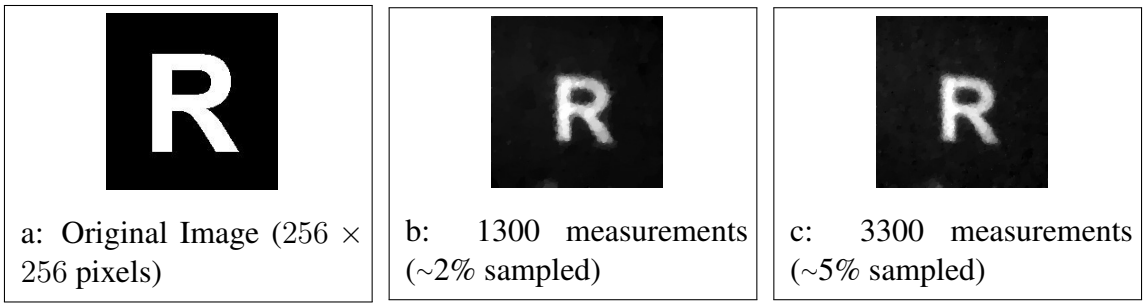


Figure 2.1: Single pixel camera for image recovery using compressed sensing. Reprinted from [2]

In this application, the single pixel camera takes thousands of samples of the image. Each sample is a projection of the light through an array of micro-mirrors and for each sample, each individual micro-mirror is randomly put into a one of two states. One state has the mirror transmit the light to light sensor and the second state does not. The configuration of the array of micro-mirrors forms a random linear combination of basis functions and the cumulative light intensity is recorded by the sensor. The cumulative light intensity and the

unique identifier (*i.e.,* random number seed) that determined the configuration of the array of micro-mirrors are the only data recorded. Using the compressed sensing methodology, as the number of samples increase, the image can be reconstructed with an increasing amount of accuracy.[1]

Figure 2.1a[2] shows an image of the letter R in a 256×256 pixel image. This is the image that will be attempted to be captured by the single pixel camera. After 1300 measurements of Figure 2.1a[2] printed and held up in front of the SPC, the results can be seen in Figure 2.1b[2]. As the measurements increase to 3300, the resolution is increased even further (see Fig. 2.1c[2]).



a: Original Image (256×256 pixels)

b: 1300 measurements (~2% sampled)

c: 3300 measurements (~5% sampled)

Figure 2.2: The original image (Figure 2.1a) and two reconstructions (Figures 2.1b and 2.1c). Reprinted from [2]

2.2 Total Variation Minimization

The problem is posed as $Ax = b$ which is an under-determined system. A is the random basis sets where each row is one basis set of $M \times N$, b is a column vector where each row is the summation of the projection of basis set onto the solution, and x is the solution,

[1] During reconstruction, the basis used for the sample is recovered from the unique identifier

[2] The discrepancy in zoom and orientations is not an error. Figure 2.1a is the original image and Figures 2.1b and 2.1c are the image printed out onto a piece of paper and held in front of the camera.

which is the unknown. With regards to the example presented in Section 2.1, each row of A is the basis recovered from the unique identifier that was recorded and the matching row in b is the cumulative light intensity. An under-determined system of equations means that x has an infinite number of solutions and therefore cannot be directly solved but the number of solutions can be reduced by imposing constraints.

We find a solution to x, through optimization, that is consistent with $Ax = b$ and minimizes some cost function designed to make x "simple". In order to find x under these conditions, we recast the problem as a second order cone problem (SOCP) and use total variation minimization (TVM) with quadratic constraints [1]. The primary difference of TVM from the more common compressed sensing technique ℓ_1 minimization, is that TVM deals with the minimization of the gradients of the data instead of minimization of the ℓ_1 norm of the data.

$$\min TV(\bar{x}) \quad \text{subject to} \quad \|Ax - b\|_2 \leq \epsilon \tag{2.1}$$

where

$$TV(\bar{x}) \equiv \sum_{ij} \sqrt{(D_{h;i,j}x)^2 + (D_{v;i,j}x)^2} \tag{2.2}$$

$$D_{h;i,j}x = \begin{cases} x_{i+1,j} - x_{i,j} & i < n \\ 0 & i = n \end{cases} \tag{2.3}$$

$D_{h;i,j}x =$ forward finite difference in the i^{th} direction

$$D_{v;i,j}x = \begin{cases} x_{i,j+1} - x_{i,j} & j < n \\ 0 & j = n \end{cases} \tag{2.4}$$

$D_{j;i,j}x =$ forward finite difference in the j^{th} direction

2.3 3D Total Variation Minimization

Modern Monte Carlo transport calculations are typically done in three dimensions. The previously defined 2-D TVM can easily be applied to a 3-D mesh by reconstruction as a series of 2-D slices. However, in some cases, it may be advantageous to minimize the gradient in higher dimensions.

Expansion of the total variation operator into a third dimension requires modifying Equation 2.2:

$$TV(\bar{x}) \equiv \sum_{ijk} \sqrt{(D_{h;i,j,k}x)^2 + (D_{v;i,j,k}x)^2 + (D_{w;i,j,k}x)^2} \tag{2.5}$$

Redefining Eqn. 2.3 and Eqn. 2.4:

$$D_{h;i,j,k}x = \begin{cases} x_{i+1,j,k} - x_{i,j,k} & i < n \\ 0 & i = n \end{cases} \tag{2.6}$$

$$D_{v;i,j,k}x = \begin{cases} x_{i,j+1,k} - x_{i,j,k} & j < n \\ 0 & j = n \end{cases} \tag{2.7}$$

and defining the forward finite difference in the k^{th} direction ($D_{w;i,j,k}x$):

$$D_{w;i,j,k}x = \begin{cases} x_{i,j,k+1} - x_{i,j,k} & k < n \\ 0 & k = n \end{cases} \tag{2.8}$$

Expansion into even higher dimensions follows the same procedure.

2.4 Log-Barrier Interior Point Method

In order to solve the TVM problem (2.1), we implement the Log-Barrier Interior Point Method. An interior point method is an alternative method to the simplex method [6], which minimizes the objective function by traversing x along the constraints. The simplex method, however, has a worst case exponential time complexity.

The core of the log-barrier interior point method is solving a series of Newton Steps. The

concept is that each of the smooth subproblems can be solved to fairly high accuracy with just a few iterations of Newton's Method. The log-barrier method is formulated by rewriting Eq. 2.1 in the form of Eq. 2.9 and incorporating the constraints into the objective function in the form of Eq. 2.11:

$$\min_x \langle c_0, x \rangle \quad \text{s.t.} \quad \begin{cases} Ax = b \\ f_i(x) \leq 0, \ i = 1, ..., m \end{cases} \tag{2.9}$$

where each f_i describes either a constraint which is linear

$$f_i = \langle c_i, x \rangle + d_i \,, \tag{2.10}$$

or second-order conic

$$f_i(x) = \frac{1}{2}(\|A_i x\|_2^2 - (\langle c_i, x \rangle + d_i)^2) \,, \tag{2.11}$$

where A_i are matrices, c_i are vectors, and d_i are scalars.

The log-barrier interior point method transforms the problem into a series of linearly constrained programs and takes the negative logarithm of the constraint slacks (introducing a parameter τ):

$$\min_x \langle c_0, x\rangle + \frac{1}{\tau^k}\sum_i -\log(-f_i(x)) \quad \text{s.t.} \quad Ax = b \tag{2.12}$$

where $\tau^k = \mu * \tau^{k-1}$ with $\mu > 0$. The parameter μ can be interpreted as a weighting factor given to the barrier function.

The choice of $-\log(-x)$ for the barrier function is chosen based on the property of being infinite when the constraint is violated or met exactly but smooth elsewhere. Under this formulation, as the solution x^k to (2.12) approaches the boundary of our feasible region, the barrier function keeps x^k within our feasible region and as τ^k gets large, x^k approaches the optimal solution to (2.9), which is denoted via x^*.

The log-barrier interior point method begins by selecting a starting point within the problem. Typically, x_0 is the solution to $Ax = b$. If this starting point is not less than the given epsilon, $x_0 = A^T w$ where w is the solution to $AA^T x = b$.

Newton's Method proceeds by forming a series of quadratic approximations to (2.9) and minimizing each by solving a system of equations. The quadratic approximation to the functional

$$f_0(x) = \langle c_0, x\rangle + \frac{1}{\tau^k}\sum_i -log(-f_i(x))\,.$$

in (2.9) around point x is

$$f_0(x + \Delta x) \approx x + \langle g_x, \Delta x \rangle + \frac{1}{2}\langle H_x \Delta x, \Delta x \rangle := q(x + \Delta x)\,, \tag{2.13}$$

where the gradient (g_x) in (2.13) is

$$g_x = c_0 + \frac{1}{\tau}\sum_i \frac{1}{-f_i(x)}\nabla^2 f_i(x)\,,$$

and the Hessian matrix (H_x) in (2.13) is

$$H_x = \frac{1}{\tau}\sum_i \frac{1}{f_i(x)^2}\nabla f_i(x)(\nabla f_i(x))^T + \frac{1}{\tau}\sum_i \frac{1}{-f_i(x)}\nabla^2 f_i(x)\,.$$

Given that x is feasible (*i.e.,* specifically $Ax = b$), the Δx that minimizes $q(x + \Delta x)$ subject to $Ax = b$ is the solution to the set of linear equations

$$\tau \begin{bmatrix} H_x & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ v \end{bmatrix} = -\tau g_x\,, \tag{2.14}$$

where vector v is the Lagrange multipliers for the quality constraints and is not directly used.

Using Δx as the Newton Step direction, the step length $s \leq 1$ is chosen such that

1. $f_i(x + s\Delta x) < 0$ for all $i = 1, \ldots, m$

2. The functional has decreased sufficiently

$$f_i(x + s\Delta x) < f_i(x) + \alpha s \Delta x \langle g_x, \Delta x \rangle \,, \tag{2.15}$$

   where α is a user-specified parameter. This requirement dictates the decrease is within a certain percentage of the predicted linear model at x.

A summary of the log-barrier algorithm is as follows:

1. Inputs: a feasible starting point x_0, a tolerance δ, and parameters μ and an initial τ^1.

2. Compute x^k by solving Equation (2.12) via Newton's Method followed by a backtracking line search using x^{k-1} as the initial point

3. If $m/\tau^k < \delta$, return x^k, else set $\tau^{k+1} = \mu\tau^k \,, k = k + 1$ and compute x^k

A full review of the log-barrier and TVM algorithms and details on the selection of τ^1 can be found in Candes and Romberg (2005) [1].

3. MONTE CARLO APPLICATION

3.1 Disjoint Tallies

The concept of random linear combinations as basis sets introduced in the Single Pixel Camera example (see Section 2.1) can be applied to scoring quantities in Monte Carlo transport. For a scoring mesh of dimensions $M \times N$:

- We define a parameter $\mathrm{r_s}$ — the subrate — which represents the fraction of the data we want to store and determines the number of basis sets we map the tally onto $\mathrm{n_B} = \mathrm{r_s} * M * N$
- We define a minimum of one random basis set and each basis set has a size of $M * N$
  - The basis set values are not required to be floating-point random from $[0, 1)$[1], and therefore, to increase the memory efficiency, we use boolean values $(\in 0, 1)$ in their most compact form: a bitset[2]
    - Efficient random generation of the basis set is done by subtracting a random 64-bit unsigned integer from the maximum possible 64-bit unsigned integer (2^{64}) and casting to a bitset
    - This is done $\frac{n-1}{64} + 1,\ n > 0$ times, where n is an integer representing the total size of the basis set and integer arithmetic is used

[1] Half-closed interval (includes 0, excludes 1). Basis set values can be floating-point random, if memory reduction is not a concern

[2] The default bitset size should be the bit-size of the CPU as this is the smallest unit of addressable memory that can be allocated (*e.g.,* a bitset of 64 should be used on a 64-bit CPU). A pair of bitsets of size 32 to represent 64 booleans would take up 2x as much memory as one bitset of 64 on a 64-bit CPU because each bitset of 32 requires a 64-bit address.

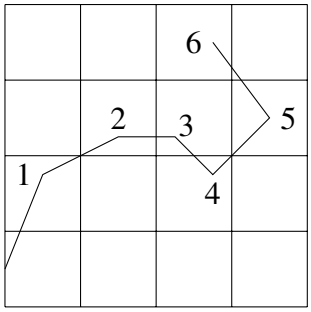


Figure 3.1: Energy Deposit Tally Illustration in 4×4 mesh

		1.5	
	0.1	0.25	1.15
0.25		0.75	

Figure 3.2: Standard Tally for Energy Deposit [keV]

An illustration of disjoint tallies is provided in Figures 3.1 through 3.6. Figure 3.1 shows a particle track through a 4×4 mesh. The particle undergoes 6 interactions — 5 scattering reactions with energy deposition and 1 absorption interaction. The values of the energy deposited are shown in Figure 3.2. For a $r_s = 0.125$, the number of disjoint tallies is equal to 2. The basis set for the disjoint tally 1 is shown in Figure 3.3 and the basis set for disjoint tally 2 is shown in Figure 3.5. The projection of the basis sets for disjoint tally 1 and 2 onto the energy deposit tally is shown in Figure 3.4 and Figure 3.6, respectively. In Figure 3.2, the energy deposited in (3, 3), where (0, 0) is the bottom left voxel, is 0.25 keV and neither basis set contains this voxel, whereas the energy deposited at (3, 4) is contained by both basis sets. The missing energy deposited is an example of where the basis sets is a factor — as the number of the basis sets increases, the probability of a voxel not being contained in any of the basis sets decreases.

0	0	1	1
1	1	0	1
0	1	0	0
1	0	1	1

Figure 3.3: Disjoint Tally Basis Set 1 of 2 for $r_s = 0.125$

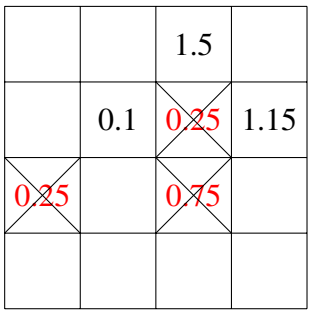


Figure 3.4: Disjoint Tally 1 for Energy Deposit. $\sum = 2.75$ keV

1	0	1	0
1	0	0	0
1	0	1	1
0	1	0	1

Figure 3.5: Disjoint Tally Basis Set 2 of 2 for $r_s = 0.125$

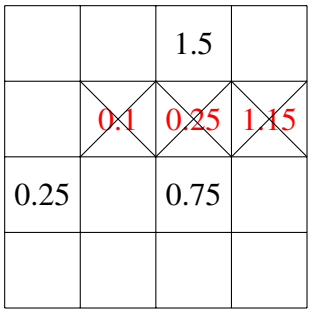


Figure 3.6: Disjoint Tally 2 for Energy Deposit. $\sum = 2.5$ keV

3.2 Monte Carlo Context

In the previous section, we introduced the disjoint tallies. The relation of these tallies to the work provided in Chapter 2 should be made clear. We stated in Section 2.2 that the total variation minimization problem is posed as $Ax = b$ but an under-determined system. In this context, the basis sets of our disjoint tallies takes the form of the matrix A, the solution to the Monte Carlo transport calculation (and the unknown) in vector form is represented by x, and the summations of the projection of the scoring quantities onto our basis sets populate the rows of the column vector b.

The disjoint basis sets (A) are randomly generated. The user can specify the random number seed provided to the random number generator used to create the disjoint basis sets[3] or else the integer representation of current time on the computer system is used. Ideally, either the specification of the random number seed for the disjoint tallies is separate (but not necessarily different) from the random number seed provided to the Monte Carlo transport code itself or the random number generator used to create A is a separate instance from the main MC random number generator. Otherwise, a variation in runtime settings of the MC code which enable the use of some portion of the code that advances the random number generator before the disjoint basis sets are generated will result in a different A even though the random number seed of the main MC random number generator may be identical.

3.3 Proof-of-Concept

The algorithm has currently been tested on an MCNP6 calculation of the Texas A&M TRIGA reactor. The MCNP6 calculation was with 200,000 particles/cycle and 2,500

[3]Specification of the same random number seed in two separate calculations will result in the same A

cycles. The reactor was meshed into 1024×1024 voxels with 32×32 sub-blocks. The fast flux (Fig. 3.7) and the thermal flux (Fig. 3.9) were tallied. We applied our compression to the solutions and reconstructed the fast flux (Fig. 3.8) and thermal flux (Fig. 3.10) from the compressed data and numerically compared our results (Fig. 3.11 and Fig. 3.12, for the fast flux and thermal flux, respectively). The total fast flux is $7.136 \times 10^{+1}$. The total difference in the solution of Fig. 3.11 is 3.338×10^{-1} with $\bar{\varepsilon} = 3.182 \times 10^{-7}$. The total thermal flux is $6.421 \times 10^{+1}$. The total difference in the solution of Fig. 3.12 is 3.347×10^{-1} with $\bar{\varepsilon} = 3.192 \times 10^{-7}$.

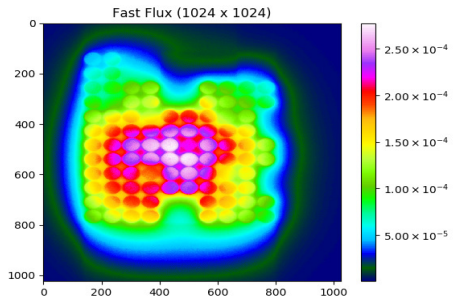


Figure 3.7: Calculated solution with standard scoring technique

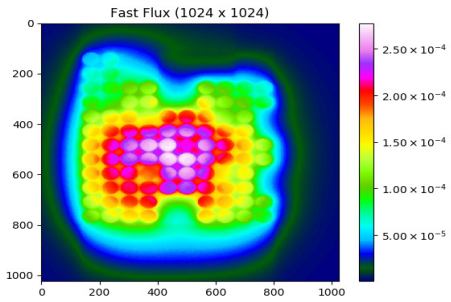


Figure 3.8: Reconstructed solution with disjoint scoring technique. $r_s = 0.5$

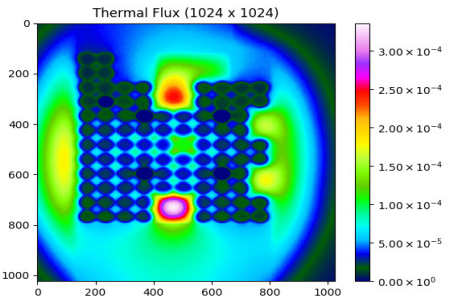


Figure 3.9: Calculated solution with standard scoring technique

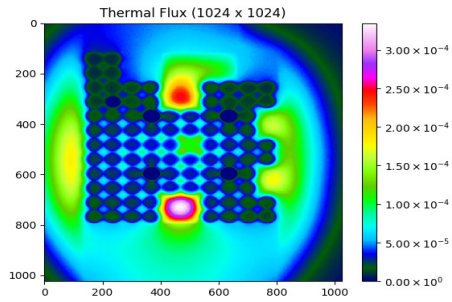


Figure 3.10: Reconstructed solution with disjoint scoring technique. $r_s = 0.5$

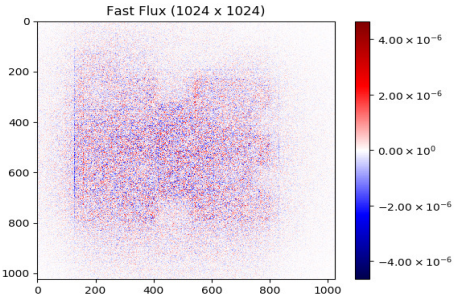


Figure 3.11: Fast flux difference between standard and disjoint scoring. $\bar{\varepsilon} = 3.182 \times 10^{-7}$

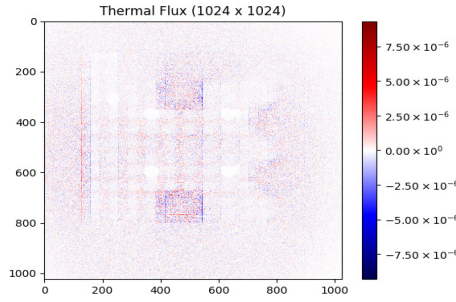


Figure 3.12: Thermal flux difference between standard and disjoint scoring. $\bar{\varepsilon} = 3.192 \times 10^{-7}$

3.4 Problem Construction

A Geant4 application was created to provide the Monte Carlo simulation and was the environment for which the reduction in runtime memory was calculated. Three independent geometries were available in this application:

1. a reactor-type geometry

2. a concrete structure shielding a uranium fuel source

3. a human phantom rendered directly from DICOM CT scans

Geant4 provides a basic tally scoring system called *Primitive Scorers* that calculate the requested value for the appropriate volume. The tally scoring was done in a Geant4 parallel world that was super-imposed over the entire world for the reactor and shielding geometries. The DICOM geometry did not require a parallel world as the geometry itself was a mesh. Appendix A provides details on the scoring fields, particle types and energies, physics lists, geometry, and materials. Instead of using the standard Geant4 scoring utility (G4THitsMap), which uses a pointer to a double-precision value keyed by an integer,[4] we implemented a more memory-efficient scoring utility of a vector of pointers. Furthermore, we did not store a lone double-precision value $\left(\sum x_i\right)$, instead we created a statistical class as follows:

- Contiguous storage of pointers to C++ class for statistical data (*e.g.,* C++ vector of pointers)

- Statistical data class contains five 64-bit fundamental data types

    1. $\sum x_i$ — running sum of estimates

    2. $\sum {x_i}^2$ — running sum of square of each estimate

    3. $\sum n_i$ — number of estimates

    4. $\sum n_{zero}$ — number of non-zero estimates

    5. $\sum w_i$ — sum of statistical weights of each estimate

[4]This data format requires 20-bytes for entry: 4 bytes for integer key, 8 bytes for pointer (on 64-bit machine) and 8 bytes for double-precision value

The 40-byte statistical data class is the minimal size in order to calculate:

- Sum
- Mean
- Efficiency
- Root-mean-squared
- Variance
- Standard Deviation
- Relative Error
- Figure-of-merit (using external timer)

These quantities are important statistical checks for Monte Carlo tallies. The G4THitsMap structure was used at the "event" level[5] and all cumulative data was stored in the aforementioned statistical data type.

3.4.1 Reactor Geometry

The reactor geometry (Figure 3.13) implemented has a cube measuring 1.5 meters on each side. The mesh for the geometry has the same dimensions as the world. There are 24 bundles of fuel pins arranged in a 5×5 configuration with one bundle removed to create asymmetry. The bundles are spaced 17.5 cm apart in X and Y directions. Each bundle has a radius of 7.5 cm. The fuel bundles consist of fuel pins with the dimensions of fuel radius of 0.386 cm and fuel height of 122.5 cm; the cladding for the fuel pins is 0.69 mm thick in

[5] An event in Geant4 is defined as the emission of one (or more) primary particles

the radial direction and 2.5 cm thick on the end caps. The number of fuel pins per bundle is 145 defined by a specification of the pins being placed in 15×15 grid where no fuel pin is created if the $\sqrt{x_{pin}^2 + y_{pin}^2} > r_{bundle}$ (the octagonal shape of fuel bundle). The pitch of the fuel pins in this configuration is 7.7 cm.

Simulations involving the reactor geometry use a random sampling of source particles: e^-, γ^0, and n_0^1. The energy of the starting particles are linearly sampled along the energy ranges: (100 keV, 10 MeV), (1 keV, 1 MeV), and (10 eV, 10 MeV), respectively.

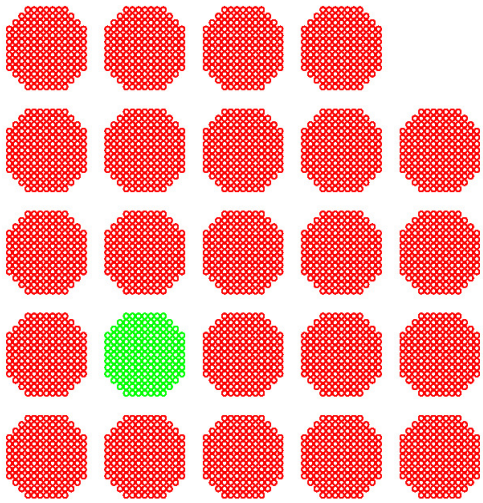


Figure 3.13: Reactor geometry. Fuel pin $\equiv$ red, Control rod $\equiv$ green

The weight-windows variance reduction technique is employed for the reactor geometry for neutrons, photons, electrons, positrons, protons, and alpha particles. The particles are indiscriminately split to a weight not exceeding 0.125 and not below 0.0625.

3.4.2 Shielding Geometry

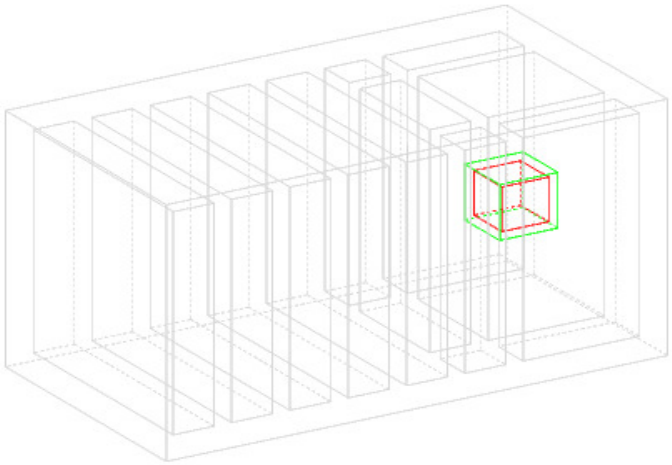


Figure 3.14: Shielding geometry. Interior cubes are air gaps, red cube is source, green cube is steel casing

The shielding geometry (Figure 3.14) followed the specification in [10] (Figure 1). A summation of this geometry is as follows:

- $460 \times 460 \times 860$ cm^3 volume
- Fuel is represented as a $80 \times 80 \times 80$ cm^3 cube of homogenized uranium dioxide and water surrounded by a 10-cm layer of steel
- Concrete walls are 30 cm thick, spaced 1 meter in each direction leaving air regions that are 70 cm thick

Simulations involving the shielding geometry use n_0^1 particles sampled from a Watt spectrum of the neutron-induced fission of U^{235}, where the starting energy of U^{235} is

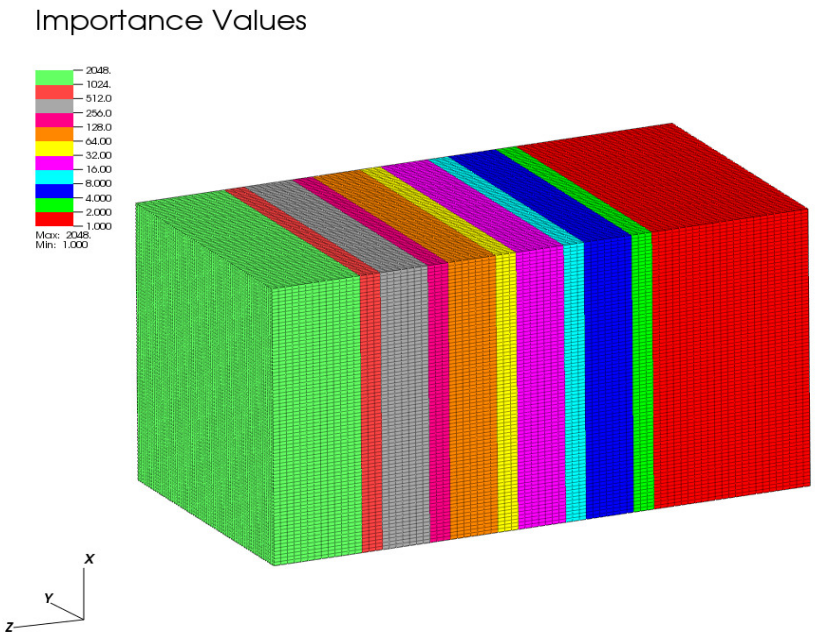


Figure 3.15: Shielding importance values

sample from a random incident energy of 1 eV to 14 MeV.

A combination of the variance reduction techniques of weight-window splitting and importance sampling are employed in the shielding geometry for neutrons and photons. The regions surrounding the source are given an importance of 1 and within the regions with this importance, no weight window variance reduction is applied. However, as the particles reach regions further away from the source, the importance values are increased by factors of 2. In these regions, weight-windows are also applied. The minimum weight of a particle in these regions does not fall below 0.0078125 (2^{-7}). The importance values are detailed in Figure 3.15.

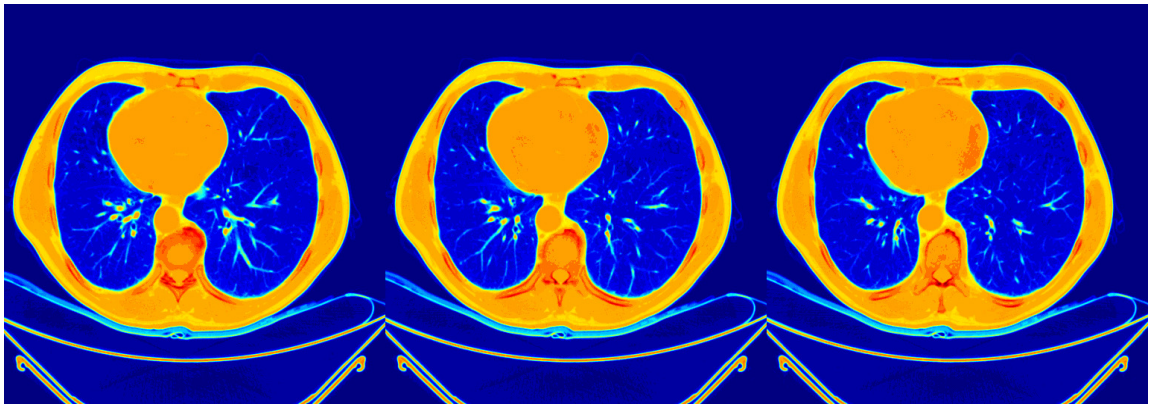


Figure 3.16: DICOM mass geometry. Slice #1: left, slice #2: center, slice #3: right

3.4.3 DICOM Geometry

The geometry in Figure 3.16 was rendered from three contiguous DICOM [6] CT scans of an anonymous patient's chest (heart, lungs, and spine). The pixel intensity of the DICOM image was mapped to densities via Hounsfield units and the material of the voxel is approximated by a correlation between standard densities of common biological materials.

Simulations involving the DICOM geometry use a source of photons linearly sampled between the energy range of 90 keV to 120 keV. The starting position of photons is randomly sampled along an imaginary cylinder surrounding the geometry with a radius equal to $\sqrt{x^2 + y^2}$ where x is the half-length of the geometry in the x-direction and y is the half-length of the geometry in the y-direction. The z-coordinate is sampled along the length of the geometry. The initial momentum direction of the source photon is sampled into a +/-27.5-degree spread with respect to the vector pointing toward the center of patient in the XY-plane.

[6] Digital Imaging and Communications in Medicine

The weight-windows variance reduction technique is employed for the DICOM geometry for neutrons, photons, electrons, positrons, protons, and alpha particles. The particles are indiscriminately split to a weight not exceeding 0.125 and not below 0.0625.

3.5 Geant4 Scorers

Geant4 has a number of primitive scorers and the following primitive scorers have been tested for their ability to be reconstructed:

- Cell Flux
- Dose Deposit
- Energy Deposit
- Population
- Number of Secondaries
- Termination
- Cell Charge
- Number of Collisions
- Number of Steps
- Passage Cell Current
- Track Length
- Flat Surface Current

- Minimum Kinetic Energy at Generation
- Passage Cell Flux
- Passage Track Length
- Track Counter

4. LIBRARY IMPLEMENTATION

The core of the library was built from the ℓ_1-magic [1] software package and translated into C++. The fast-flux proof-of-concept problem in Section 3.3 required an excess of three days to calculate serially in MATLAB on an Intel Core i7 CPU @ 3.50GHz. The C++ implementation required 57 minutes using 4 threads on an Intel Core i7 @ 2.50GHz with a NVIDIA 940MX GPU (512 CUDA cores).

4.1 Package Overview

4.1.1 Dependencies

The aforementioned algorithms defined in Sections 2.2, 2.3, and 2.4 were implemented as C++11 library with dependencies on the Armadillo linear algebra library (version 7.800.1+) [11], the Boost serialization library [12], and CMake (version 3.3+) [13] for compilation. The library handles both the proof-of-concept case (full, known solution with post-compression and reconstruction) and the runtime compression case where the data is provided to the library in the compressed form. Compilation with the Dlib library [14] provides a utility to produce images of the solutions.

If the SWIG package [15] is available, a Python interface is available to construct problems. If the ArmaNpy [16] package is available, in addition to SWIG, then utilities are provided to utilize Python's plotting utilities as an improved alternative to the C++ imaging through Dlib or the standalone bitmap implementation. All heat-map plots provided in this dissertation with data ranges were generated using Python and the remaining heat-map plots were generated with Dlib.

4.1.2 Parallelism

On UNIX systems, the library uses POSIX thread (pthreads) in a task-based, work-stealing thread pool for shared memory parallelism where each task is a defined by a sub-block unit for reconstruction. Speed-up is nearly linear (~0.90):

$$\frac{T_1}{N * T_N} \simeq 1 \quad \text{where N is number of threads}$$

Evidence of this can be found in Section 4.4 where reconstructions with 16 threads demonstrate a speed-up factor around an average of 14.5 according to the fraction of the CPU time divided by the wall clock time.[1]

The Armadillo library allows easy use of GPU acceleration by linking with the NVIDIA CUDA NVBLAS library and/or AMD ACML-GPU library. There is also automatic SIMD vectorization support and high-speed BLAS replacements for matrix multiplication and matrix inversion can be utilized by linking to OpenBLAS or Intel MKL libraries (and the Accelerate framework on Mac OS X). The SuperLU library [17] is required for solving sparse systems. It is highly-recommended to utilize the Intel MKL library when available.[2]

4.2 Monte Carlo Compute Time

The wall clock and CPU time of several simulations were recorded for all three geometries detailed in Section 3.4. These simulations vary in subrate, number of particles, and number

[1] Ideal speed-up on 16 threads is 16 but can be exceeded when combined with vectorization

[2] It has been noted on several machines that the Intel MKL library improves certain parts of the code, specifically, when finding a suitable starting point in the log-barrier algorithm

of tallies. All simulations in this section utilized 16 threads on four quad-core Intel Xeon CPUs @ 2.27 GHz.

The intricacies of the Geant4 scoring method should be noted here. Before that can be done, we must define the following Monte Carlo simulation terms:

- Step — Smallest unit of simulation. It may be a process the particle undergoes or a transport step
- Track — The collection of steps spanning the creation of the particle until termination at zero kinetic energy, absorption, or exiting "world" volume
- Event — The collection that begins with one or more primary tracks and includes all of the secondary particles generated as a result of transporting the primary particles
- Run — The total collection of events that encapsulate the simulation. This may be thought of as n, where n is the number of samples described in Section 1.4

For each event in Geant4, which in the simulations of this dissertation is one primary particle, an "event map" containing all the tally values are keyed to the unique ID of the voxel. At the conclusion of the event, this event map is passed to the "run tallies" for accumulation over all the events (a run), which is either the standard tallies (in the form of a vector) or the disjoint tallies. When Geant4 is compiled with multithreading, the run tallies for each worker thread maintain their own copy (*i.e.,* thread-local) and each worker thread-local copy of the tallies is merged into a master thread-local copy at the conclusion of the run. In other words, if 16 threads are utilized, 17 copies of the tallies are created (1 master + 16 worker) and each worker updates only their own copy of the tally until the very end of the simulation when all 16 worker copies are merged into the single master copy.

With regards to the transfer from the event map to the run tallies, recording the event map entries in standard tallies requires only one operation for each entry in the map, which is adding the value at the map key to the value at the vector entry at the index of the map key. However, recording the event map for disjoint tallies requires n_B operations, where n_B is the number of basis sets defined in Section 3.1. Consequently, there are three primary factors that can affect performance when using disjoint tallies.

1. The frequency at which the disjoint tallies are updated
2. The size of the event map during the update
3. The number of basis sets

An increase in any of these areas will negatively impact performance because it is increasing the total number of operations that must be done to update the disjoint tallies while the number of operations required by standard tallies remains constant. However, for most simulations, this increase is mostly negligible and the creation of the intermediate event map works to the advantage of the disjoint tallies versus direct update by every sample. In general, particles with large mean free paths (MFP) in finely discretized meshes suffer the most significant performance losses due to the combination of size of the intermediate event map (*i.e.,* many voxels were updated even if these values were mostly negligible) and the short computation time of the single event (*i.e.,* only a couple interactions and relatively few secondary particles).

4.2.1 Reactor Problem Compute Time

Timing results for the reactor geometry with two tallies (energy deposit and minimum kinetic energy at generation) can be found in Table 4.1 and the CPU utilization can be

found in Table 4.2.

Table 4.1: Simulation times [minutes] for $512 \times 512 \times 2$ reactor geometry using $32 \times 32 \times 1$ sub-blocks

# event	r_s	# tally	$Wall_S$	$Wall_D$	Δ_{Wall}	CPU_S	CPU_D	Δ_{CPU}
2.5×10^5	0.25	15	169.8	176.2	3.8%	2474.8	2528.8	2.2%
2.5×10^5	0.50	15	169.4	176.6	4.3%	2430.8	2609.1	7.3%
2.5×10^5	0.75	15	166.9	177.4	6.3%	2457.7	2604.0	6.0%

Table 4.2: % CPU for $512 \times 512 \times 2$ reactor geometry using $32 \times 32 \times 1$ sub-blocks

# event	r_s	# tally	% CPU_S	% CPU_D
2.5×10^5	0.25	15	1458%	1435%
2.5×10^5	0.50	15	1435%	1477%
2.5×10^5	0.75	15	1473%	1467%

4.2.2 Shielding Problem Compute Time

Timing results for the shielding geometry with a mesh of $40 \times 40 \times 80$ scoring two tallies (cell flux and dose deposit) and four tallies (cell flux, dose deposit, cell charge, and termination) can be found in Tables 4.3 and 4.5 in sub-blocks of $20 \times 20 \times 2$ and $40 \times 40 \times 1$, respectively. The corresponding CPU utilization can be found in Tables 4.4 and 4.6.

Table 4.3: Simulation times [minutes] for $40\times40\times80$ shielding geometry using $20\times20\times2$ sub-blocks

# event	r_s	# tally	$Wall_S$	$Wall_D$	Δ_{Wall}	CPU_S	CPU_D	Δ_{CPU}
1×10^5	0.25	2	7.2	7.4	2.3%	99.3	101.5	2.2%
1×10^5	0.50	2	7.2	7.4	1.7%	99.8	101.5	1.7%
1×10^5	0.75	2	7.2	7.5	4.7%	98.9	104.2	5.3%
5×10^5	0.25	2	36.5	37.1	1.7%	505.6	513.1	1.5%
5×10^5	0.50	2	36.3	36.7	1.0%	503.3	506.7	0.7%
5×10^5	0.75	2	35.9	37.7	5.0%	500.1	520.6	4.1%
1×10^5	0.25	4	7.4	7.5	1.4%	101.2	103.3	2.1%
1×10^5	0.50	4	7.6	7.5	-1.5%	103.9	103.3	-0.6%
1×10^5	0.75	4	7.3	7.5	2.1%	100.9	103.5	2.6%
5×10^5	0.25	4	36.3	37.5	3.1%	500.6	517.3	3.3%
5×10^5	0.50	4	36.8	37.7	2.4%	508.0	522.5	2.8%
5×10^5	0.75	4	36.0	37.9	5.3%	498.1	522.2	4.8%

Table 4.4: % CPU for $40 \times 40 \times 80$ shielding geometry using $20 \times 20 \times 2$ sub-blocks

# event	r_s	# tally	% CPU_S	% CPU_D
1×10^5	0.25	2	1377%	1376%
1×10^5	0.50	2	1378%	1379%
1×10^5	0.75	2	1372%	1380%
5×10^5	0.25	2	1387%	1384%
5×10^5	0.50	2	1385%	1381%
5×10^5	0.75	2	1393%	1381%
1×10^5	0.25	4	1371%	1381%
1×10^5	0.50	4	1374%	1386%
1×10^5	0.75	4	1376%	1383%
5×10^5	0.25	4	1378%	1381%
5×10^5	0.50	4	1380%	1386%
5×10^5	0.75	4	1385%	1379%

Table 4.5: Simulation times [minutes] for $40\times40\times80$ shielding geometry using $40\times40\times1$ sub-blocks

# event	r_s	# tally	$Wall_S$	$Wall_D$	Δ_{Wall}	CPU_S	CPU_D	Δ_{CPU}
1×10^5	0.25	2	7.3	7.5	2.8%	101.0	103.1	2.1%
1×10^5	0.50	2	7.3	7.5	2.8%	101.3	103.2	1.9%
1×10^5	0.75	2	7.2	7.6	5.7%	98.5	104.5	6.1%
5×10^5	0.25	2	36.9	37.5	1.7%	505.0	513.4	1.7%
5×10^5	0.50	2	36.7	37.6	2.4%	503.3	515.5	2.4%
5×10^5	0.75	2	37.1	38.5	3.9%	510.5	527.6	3.4%
1×10^5	0.25	4	7.3	7.5	2.0%	101.4	103.2	1.8%
1×10^5	0.50	4	7.4	7.6	2.4%	101.4	104.6	3.2%
1×10^5	0.75	4	7.3	7.6	4.2%	99.8	105.1	5.3%
5×10^5	0.25	4	37.3	38.2	2.4%	511.2	524.0	2.5%
5×10^5	0.50	4	37.5	38.6	2.8%	513.1	529.1	3.1%
5×10^5	0.75	4	37.4	38.9	4.2%	511.1	533.7	4.4%

Table 4.6: % CPU for $40 \times 40 \times 80$ shielding geometry using $40 \times 40 \times 1$ sub-blocks

# event	r_s	# tally	% CPU_S	% CPU_D
1×10^5	0.25	2	1392%	1382%
1×10^5	0.50	2	1382%	1370%
1×10^5	0.75	2	1372%	1377%
5×10^5	0.25	2	1368%	1368%
5×10^5	0.50	2	1370%	1370%
5×10^5	0.75	2	1375%	1369%
1×10^5	0.25	4	1383%	1381%
1×10^5	0.50	4	1367%	1377%
1×10^5	0.75	4	1368%	1382%
5×10^5	0.25	4	1370%	1372%
5×10^5	0.50	4	1367%	1371%
5×10^5	0.75	4	1368%	1371%

4.2.3 DICOM Problem Compute Time

Timing results for the DICOM geometry with two tallies (energy deposit and minimum kinetic energy at generation) can be found in Table 4.7 and the CPU utilization can be found in Table 4.8 where a significant performance loss can be noted. This DICOM geometry is an example of a finely discretized mesh with the primary particles (γ) having large MFP. However, this performance hit can be offset by reducing the size of the basis sets, as seen in Table 4.9. Although, reducing the size of the basis sets will have a negative impact on the reconstruction quality if taken to the extreme. The reduction in reconstruction quality will largely occur at the boundaries of sub-domain and manifest itself in the form of discontinuities between adjacent sub-domains. The sub-domain in Table 4.9 has the minimum recommended size of 16×16.

As noted in the last paragraph, the DICOM geometry suffers significantly from the implementation of disjoint tallies, but this is in contrast to the results seen later in Sections 4.2.1 and 4.2.2, which have mostly negligible increases in runtime even with the same mesh and sub-domain specifications as the DICOM geometry. With this in mind, a puzzling part of the DICOM simulation can be seen in Tables 4.8 and 4.10. Despite running on 16 threads, both the disjoint and standard tallies display below average CPU utilization. The disparity in the runtime of the standard tallies vs. the disjoint tallies has been covered but this does not explain the poor CPU utilization of the standard tallies, since the tallies are thread-local. The cause has not been determined, although is likely attributed to a blocking mechanism or false-sharing[3] within the Geant4 kernel itself and it may exacerbate the disjoint tally update time addition.

[3]False-sharing is when 2 or more threads allocate data on the same cache line and an indirect blocking event occurs while the threads constantly swap updating the cache line

Table 4.7: Simulation times [minutes] for $512\times512\times2$ DICOM geometry using $32\times32\times1$ sub-blocks

# event	r_s	# tally	$Wall_S$	$Wall_D$	Δ_{Wall}	CPU_S	CPU_D	Δ_{CPU}
1×10^6	0.30	4	6.0	9.3	55.8%	51.7	109.9	112.6%
1×10^6	0.50	4	6.1	11.3	87.0%	51.6	139.1	169.6%
1×10^6	0.70	4	5.8	12.7	118.3%	51.6	158.6	207.4%
1×10^7	0.30	4	63.0	100.4	59.4%	517.2	1101.3	113.0%
1×10^7	0.50	4	64.1	118.4	84.8%	514.9	1370.9	166.2%
1×10^7	0.70	4	64.7	143.9	122.5%	518.8	1623.3	212.9%

Table 4.8: % CPU for $512 \times 512 \times 2$ DICOM geometry using $32 \times 32 \times 1$ sub-blocks

# event	r_s	# tally	% CPU_S	% CPU_D
1×10^6	0.30	4	864%	1179%
1×10^6	0.50	4	851%	1227%
1×10^6	0.70	4	885%	1247%
1×10^7	0.30	4	821%	1097%
1×10^7	0.50	4	804%	1158%
1×10^7	0.70	4	802%	1128%

Table 4.9: Simulation times [minutes] for $512\times512\times2$ DICOM geometry using $16\times16\times1$ sub-blocks

# particles	r_s	# tally	$Wall_S$	$Wall_D$	Δ_{Wall}	CPU_S	CPU_D	Δ_{CPU}
1×10^6	0.30	4	7.3	6.5	-10.4%	51.9	67.1	29.3%
1×10^6	0.50	4	7.2	7.1	-0.6%	50.4	75.5	49.8%
1×10^6	0.70	4	6.7	8.5	26.5%	51.6	85.3	65.2%
1×10^7	0.30	4	56.6	66.8	17.9%	505.0	669.4	32.6%
1×10^7	0.50	4	58.7	81.5	38.9%	512.6	766.4	49.5%
1×10^7	0.70	4	57.0	79.9	40.2%	512.0	846.8	65.4%

Table 4.10: % CPU for $512 \times 512 \times 2$ DICOM geometry using $16 \times 16 \times 1$ sub-blocks

# particles	r_s	# tally	% CPU_S	% CPU_D
1×10^6	0.30	4	714%	1031%
1×10^6	0.50	4	705%	1062%
1×10^6	0.70	4	766%	1000%
1×10^7	0.30	4	892%	1003%
1×10^7	0.50	4	874%	941%
1×10^7	0.70	4	899%	1060%

4.3 Reconstruction Parameters

The quality of the reconstruction is determined from four primary parameters:

1. Subrate — r_s
2. Epsilon — ε
3. Iteration tolerance — δ
4. Barrier factor — μ

The most significant parameter is the subrate. As described in Section 3.1, the subrate determines the basis sets for the reconstruction. A larger subrate setting will result in a more precise reconstruction with respect to the standard tallies. A lower subrate setting will result in an increase of statistical denoising, but too low of a subrate setting in this area can lead to a significant loss of detail in reconstructions that are diverse in their material distributions, *e.g.,* the DICOM problem. The subrate is also proportional to the memory savings and the recommended subrate, based on our experience, that provides the best balance of memory reduction, statistical denoising, and inclusion of detail is 0.5.

The remaining three parameters are only used during the actual reconstruction and have no effect on memory reduction but will affect the reconstruction time. Epsilon is the same epsilon found in Equation 2.1 and determines the constraint to which the solution for the minimization of the total variation satisfies $\|\mathrm{A}x - \mathrm{b}\|_2$. A larger value of epsilon will increase the statistical denoising. The recommended value, based on our experience, for epsilon is 1×10^{-3} for standard denoising and 1×10^{-2} for increased denoising.

The iteration tolerance ε is used when calculating the number of log-barrier iterations and as the tolerance to terminate the Newton step(s), *i.e.,* terminate the Newton step(s) when the Newton decrement falls below this value. The number of log-barrier iterations scales with $-\log \delta$ and the recommended value is ε or within the range of $[1 \times 10^{-2} * \varepsilon,\ \varepsilon]$.

The final parameter μ is the factor by which to increase the barrier constant at each iteration. A larger value of μ implies a larger step towards the boundary of the feasible region and typical values are as low as 2 and as high as 20. The number of log-barrier iterations is inversely proportional to $\log \mu$ and values in the range of $[10,\ 20]$ is recommended, based on our experience, for problems without a high variance in material distributions and values in the range of $[2,\ 5]$ for problems with a high degree of material variance (*e.g.,* DICOM problem).

4.4 Reconstruction Compute Time

All reconstruction timings listed reconstructed the $\sum x_i$, $\sum {x_i}^2$, and $\sum n_i$. Approximate timing for the $\sum x_i$ alone are $^1/_3$ of the stated reconstruction time.

4.4.1 DICOM Problem Reconstruction

The reconstruction parameters for the DICOM geometry are found in Tables 4.11 and 4.12. The DICOM geometry was reconstructed at two different iteration tolerances to demonstrate the dramatic effect on computation time that can occur in some geometries, specifically ones that are materially diverse.

Table 4.11: Reconstruction parameters set A for the DICOM geometry

ε	1×10^{-3}
δ	1×10^{-4}
μ	2

Table 4.12: Reconstruction parameters set B for the DICOM geometry

ε	1×10^{-3}
δ	1×10^{-3}
μ	10

Table 4.13: Reconstruction time [minutes] for $512 \times 512 \times 2$ DICOM geometry using $32 \times 32 \times 1$ sub-blocks with parameters from Table 4.11

# events	r_s	Tally	$Wall_D$	CPU_D	% CPU_D
1×10^7	0.30	MinKinEAtGeneration	114.9	1725.0	1501.8%
1×10^7	0.30	NumOfCollision	110.0	1664.7	1513.2%
1×10^7	0.30	Termination	110.0	1644.8	1494.7%
1×10^7	0.30	TrackLength	97.9	1457.1	1488.0%
1×10^7	0.50	MinKinEAtGeneration	116.9	1773.6	1517.6%
1×10^7	0.50	NumOfCollision	113.7	1724.8	1517.2%
1×10^7	0.50	Termination	112.4	1704.6	1516.9%
1×10^7	0.50	TrackLength	92.8	1394.9	1503.8%
1×10^7	0.70	MinKinEAtGeneration	122.4	1858.4	1518.1%
1×10^7	0.70	NumOfCollision	116.2	1762.2	1516.6%
1×10^7	0.70	Termination	116.0	1759.9	1517.1%
1×10^7	0.70	TrackLength	95.3	1436.8	1508.4%

Table 4.14: Reconstruction time [minutes] for $512 \times 512 \times 2$ DICOM geometry using $32 \times 32 \times 1$ sub-blocks with parameters from Table 4.12

# events	r_s	Tally	$Wall_D$	CPU_D	% CPU_D
1×10^7	0.30	MinKinEAtGeneration	38.1	573.6	1505.2%
1×10^7	0.30	NumOfCollision	37.6	564.8	1502.8%
1×10^7	0.30	Termination	35.5	532.6	1499.2%
1×10^7	0.30	TrackLength	34.0	509.3	1499.9%
1×10^7	0.50	MinKinEAtGeneration	38.9	590.5	1517.2%
1×10^7	0.50	NumOfCollision	38.6	584.3	1514.8%
1×10^7	0.50	Termination	36.1	547.4	1517.0%
1×10^7	0.50	TrackLength	32.0	471.1	1472.8%
1×10^7	0.70	MinKinEAtGeneration	39.7	601.7	1516.5%
1×10^7	0.70	NumOfCollision	40.0	606.1	1516.0%
1×10^7	0.70	Termination	38.8	587.9	1516.9%
1×10^7	0.70	TrackLength	32.7	492.3	1506.6%

4.4.2 Reactor Problem Reconstruction

The reconstruction parameters for the reactor-type geometry can be found in Table 4.15. For this geometry, reconstructions were performed on all of the Geant4 scorers from Section 3.5 with the exception of the minimum kinetic energy at generation. The results are separated by subrate into Table 4.16 for $r_s = 0.25$, Table 4.17 for $r_s = 0.50$, and Table 4.18 for $r_s = 0.75$.

Table 4.15: Reconstruction parameters for reactor-type geometry

ε	1×10^{-3}
δ	1×10^{-3}
μ	5

Table 4.16: Reconstruction times [minutes] for $512 \times 512 \times 2$ reactor-type geometry using $32 \times 32 \times 1$ sub-blocks @ $r_s = 0.25$

# event	r_s	Tally	$Wall_D$	CPU_D	% CPU_D
1×10^5	0.25	CellCharge	16.7	232.3	1393.7%
1×10^5	0.25	CellFlux	11.3	152.5	1345.4%
1×10^5	0.25	DoseDeposit	12.8	174.1	1364.6%
1×10^5	0.25	EnergyDeposit	12.6	172.8	1366.7%
1×10^5	0.25	FlatSurfaceCurrent	13.9	192.2	1380.9%
1×10^5	0.25	NumOfCollision	11.7	157.8	1350.4%
1×10^5	0.25	NumOfSecondary	12.7	172.9	1362.3%
1×10^5	0.25	NumStep	12.0	163.0	1356.3%
1×10^5	0.25	PassageCellCurrent	11.1	149.0	1345.1%
1×10^5	0.25	PassageCellFlux	11.4	153.6	1349.0%
1×10^5	0.25	PassageTrackLength	11.3	151.7	1347.2%
1×10^5	0.25	Population	11.2	150.4	1343.6%
1×10^5	0.25	Termination	13.1	179.1	1367.7%
1×10^5	0.25	TrackCounter	11.9	160.5	1354.0%
1×10^5	0.25	TrackLength	11.6	156.5	1346.8%

Table 4.17: Reconstruction times [minutes] for $512 \times 512 \times 2$ reactor-type geometry using $32 \times 32 \times 1$ sub-blocks @ $r_s = 0.50$

# event	r_s	Tally	$Wall_D$	CPU_D	% CPU_D
1×10^5	0.50	CellCharge	16.4	237.3	1444.2%
1×10^5	0.50	CellFlux	11.3	157.4	1394.3%
1×10^5	0.50	DoseDeposit	12.2	171.2	1407.5%
1×10^5	0.50	EnergyDeposit	12.1	170.3	1403.5%
1×10^5	0.50	FlatSurfaceCurrent	14.3	204.7	1434.3%
1×10^5	0.50	NumOfCollision	12.3	173.9	1410.1%
1×10^5	0.50	NumOfSecondary	13.0	184.8	1416.9%
1×10^5	0.50	NumStep	12.1	169.6	1401.8%
1×10^5	0.50	PassageCellCurrent	11.6	162.2	1396.8%
1×10^5	0.50	PassageCellFlux	11.7	163.2	1398.9%
1×10^5	0.50	PassageTrackLength	11.6	162.7	1398.7%
1×10^5	0.50	Population	11.6	162.7	1398.4%
1×10^5	0.50	Termination	13.5	192.6	1424.0%
1×10^5	0.50	TrackCounter	12.0	169.5	1407.2%
1×10^5	0.50	TrackLength	11.7	164.0	1401.9%

Table 4.18: Reconstruction times [minutes] for $512 \times 512 \times 2$ reactor-type geometry using $32 \times 32 \times 1$ sub-blocks @ $r_s = 0.75$

# event	r_s	Tally	$Wall_D$	CPU_D	% CPU_D
1×10^5	0.75	CellCharge	18.7	261.8	1402.7%
1×10^5	0.75	CellFlux	12.7	172.0	1357.1%
1×10^5	0.75	DoseDeposit	13.5	184.8	1370.5%
1×10^5	0.75	EnergyDeposit	13.2	179.9	1364.8%
1×10^5	0.75	FlatSurfaceCurrent	16.2	226.4	1396.0%
1×10^5	0.75	NumOfCollision	14.2	196.1	1377.5%
1×10^5	0.75	NumOfSecondary	14.6	201.2	1381.5%
1×10^5	0.75	NumStep	13.6	187.4	1373.9%
1×10^5	0.75	PassageCellCurrent	12.8	175.1	1364.4%
1×10^5	0.75	PassageCellFlux	13.3	181.0	1364.4%
1×10^5	0.75	PassageTrackLength	13.2	180.3	1365.8%
1×10^5	0.75	Population	13.0	176.7	1363.2%
1×10^5	0.75	Termination	15.1	209.4	1383.3%
1×10^5	0.75	TrackCounter	13.7	188.2	1370.0%
1×10^5	0.75	TrackLength	13.2	180.0	1365.1%

4.4.3 Shielding Problem Reconstruction

The reconstruction parameters for the shielding geometry can be found in Table 4.19. Reconstructions for two separate sub-block configurations are provided in Tables 4.3 and 4.5. The sub-block configurations are $20 \times 20 \times 2$ and $40 \times 40 \times 1$ within the $40 \times 40 \times 80$ mesh. Reconstructions for four tallies are provided for three subrates at 1×10^5 and 5×10^5 events.

Table 4.19: Reconstruction parameters for shielding geometry

ε	1×10^{-3}
δ	1×10^{-3}
μ	10

Table 4.20: Reconstruction time [minutes] for $40 \times 40 \times 80$ shielding geometry using $20 \times 20 \times 2$ sub-blocks

# event	r_s	Tally	$Wall_D$	CPU_D	% CPU_D
1×10^5	0.25	CellCharge	6.8	103.7	1515.6%
1×10^5	0.25	CellFlux	6.9	105.2	1519.0%
1×10^5	0.25	DoseDeposit	7.3	110.4	1508.8%
1×10^5	0.25	Termination	9.3	138.2	1493.4%
5×10^5	0.25	CellCharge	7.0	106.0	1513.9%
5×10^5	0.25	CellFlux	6.0	90.8	1512.8%
5×10^5	0.25	DoseDeposit	7.7	115.8	1511.6%
5×10^5	0.25	Termination	9.6	146.4	1518.1%
1×10^5	0.50	CellCharge	7.7	117.2	1517.5%
1×10^5	0.50	CellFlux	7.9	119.6	1515.7%
1×10^5	0.50	DoseDeposit	7.6	114.8	1503.1%
1×10^5	0.50	Termination	9.2	138.9	1510.5%
5×10^5	0.50	CellCharge	7.4	111.5	1512.1%
5×10^5	0.50	CellFlux	6.8	103.1	1517.2%
5×10^5	0.50	DoseDeposit	8.5	128.2	1507.1%
5×10^5	0.50	Termination	10.8	159.9	1480.4%
1×10^5	0.75	CellCharge	8.4	126.1	1508.5%
1×10^5	0.75	CellFlux	8.7	131.8	1508.5%
1×10^5	0.75	DoseDeposit	8.5	126.5	1479.7%
1×10^5	0.75	Termination	10.2	154.7	1509.4%
5×10^5	0.75	CellCharge	8.2	123.0	1503.9%
5×10^5	0.75	CellFlux	7.3	110.5	1509.6%
5×10^5	0.75	DoseDeposit	9.2	136.7	1490.9%
5×10^5	0.75	Termination	11.0	165.1	1500.9%

Table 4.21: Reconstruction time [minutes] for $40 \times 40 \times 80$ shielding geometry using $40 \times 40 \times 1$ sub-blocks

# event	r_s	Tally	$Wall_D$	CPU_D	% CPU_D
5×10^5	0.25	CellCharge	10.3	153.3	1486.5%
5×10^5	0.25	CellFlux	8.1	122.4	1511.0%
5×10^5	0.25	DoseDeposit	12.1	178.7	1482.2%
5×10^5	0.25	Termination	13.0	195.9	1509.1%
5×10^5	0.50	CellCharge	10.5	166.2	1586.0%
5×10^5	0.50	CellFlux	9.1	144.4	1591.9%
5×10^5	0.50	DoseDeposit	12.2	192.9	1587.6%
5×10^5	0.50	Termination	14.1	220.7	1569.0%
5×10^5	0.75	CellCharge	11.6	176.6	1525.0%
5×10^5	0.75	CellFlux	10.5	160.1	1522.1%
5×10^5	0.75	DoseDeposit	13.9	207.4	1490.3%
5×10^5	0.75	Termination	15.5	235.5	1522.9%

4.5 Variance Reconstruction

For our compressed sensing algorithm, we store the $\sum x_i^2$ and N in the same method that we store the $\sum x_i$ to enable reconstruction of the variance. In addition, the root mean squared (Equation 1.5), the relative error (Equation 1.6), and the coefficient of variation (Equation 1.7) are available for each reconstruction.

However, the accuracy of these quantities suffer from a minor problem: the n quantity is reconstructed as a floating-point value and any attempts to cast this value to an integer type, as it would be found in standard scoring, would possibly invalidate the conservation. However, reconstructions in Appendix B.2.2 demonstrate that the magnitude of the relative error, variance, RMS, and coefficient of variation are on par with the reported values from standard scoring. Furthermore, we have found that for quantities that are inherently greater than zero (*e.g.,* energy deposit, flux, number of secondaries, number of collisions, minimum kinetic energy at generation, etc.), noisy reconstructions (*i.e.,* reconstructions with under-sampled voxels) will report negative numbers — providing the disjoint tally methodology with its own statistical test. Certain tolerance and Newton-stepping parameters are available to reduce the occurrence likelihood of these negative values[4] but a full reconstruction without negative solutions cannot be guaranteed in this case when the solution is overly noisy. However, when these negative values are found and the reconstruction field has been specified as non-negative, these values by default will set to zero and an output of the negative values is provided separately. Future iterations of the library will incorporate constraints into the TVM reconstruction to enable finding solutions with all values greater than zero.

[4] A routine has been placed in the reconstruction library to enable trying a second minimization search with modified parameters when the reconstruction field has been deemed as globally non-negative and a negative value is found

4.6 Conservation of Reconstructed Quantities

We store a total for all these quantities for the entire sub-domain to scale the solutions of our reconstructions and ensure conservation. The total for the entire domain is utilized at the end of the reconstruction to ensure total conservation. Prior to the reconstruction calculation, each sub-domain is normalized and the local total for the sub-domain is used to scale the sub-domain after the reconstruction. The localized scaling of each sub-domain is necessary to reduce boundary discontinuities between the sub-domains.

5. RESULTS

5.1 Comparison Methodology

In order to demonstrate the validity of our solutions when comparing a standard tally with a disjoint tally featuring the statistical denoising, we must establish a measurement for quantifying the difference between our solutions. This measurement should contain the following properties:

1. Decrease as the number of estimates of the value(s) increase
2. Take all statistical fluctuations into account
3. Locally quantifiable
4. Represented as a single value

A measurement containing all the properties is the aforementioned total variation. If we define Equation 2.2 and Equation 2.5 as the isotropic total variation for 2- and 3-dimensions, we can also define the 2D anisotropic total variation (omitting the straight-forward expansion for 3 dimensions) as:

$$TV_{aniso}(\bar{x}) \equiv \sum_{ij}^{N} |x_{i+1,j} - x_{i,j}| + \sum_{ij}^{N} |x_{i,j+1} - x_{i,j}| \tag{5.1}$$

These two measurements, the isotropic and anisotropic total variation, will form the basis of our comparison for the disjoint tally method with the standard tally method. Analysis

of Equation 2.2 and 5.1 will show that requirements (2), (3), and (4) are satisfied.

In order to demonstrate that the total variation of global Monte Carlo tallies are inversely proportional to the number of particles that are simulated (condition 1), we chose three tallies:

1. Cell Flux

2. Energy Deposit

3. Number of Secondaries

These three tallies were chosen for their relative diversity in the physics resulting in the tally being recorded: cell flux is a highly dependent on the particle source, energy deposition is highly dependent on the properties of the particle itself (*e.g.,* charge, kinetic energy, etc.), and the number of secondaries is highly dependent on the material. Although this is a very broad generalization, there is no direct correlation between these three tallies and their magnitudes tend to differ significantly so we posit these three tallies alone will suffice as a demonstration of the inverse proportionality of total variation for all global Monte Carlo tallies.

Several simulations with increasing amounts of primary particles were calculated and the isotropic and anisotropic total variation of both the standard tallies and the disjoint tallies were recorded. All simulations were started with different random number seeds.

Figure 5.1 and 5.2 demonstrate that not only are the standard and disjoint tallies reducing their total variation as the number of particle are increased but that our disjoint tallies are converging towards the same ideal total variation as the standard tallies as the number of particles goes to infinity for both the isotropic and anisotropic formulations. This property

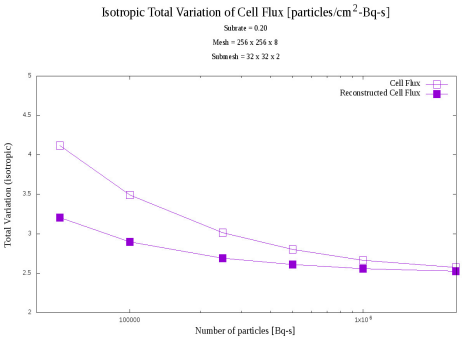


Figure 5.1: Total Variation (ℓ_{iso}) of Mesh Tally for Cell Flux ($r_s = 0.2$)

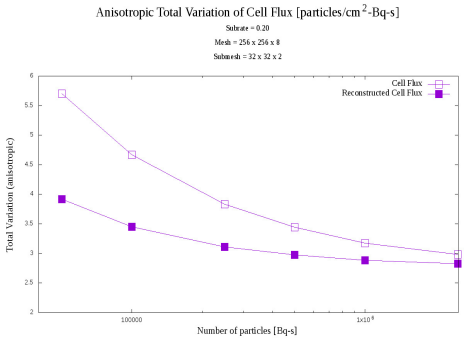


Figure 5.2: Total Variation (ℓ_{aniso}) of Mesh Tally for Cell Flux ($r_s = 0.2$)

is reproduced in Figure B.1 and B.2 (energy deposit) and Figure B.3 and B.4 (number of secondaries) in Appendix B. Later on, we will compare the standard tallies and the disjoint tallies to higher resolved solutions to demonstrate the validity of the statistical denoising.

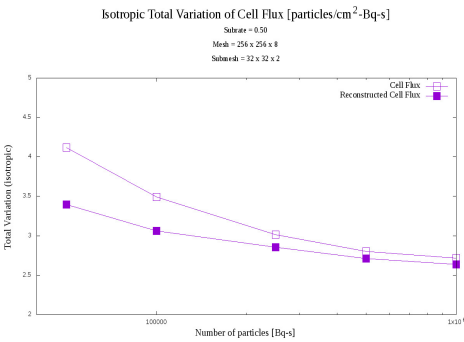


Figure 5.3: Total Variation (ℓ_{iso}) of Mesh Tally for Cell Flux ($r_s = 0.5$)

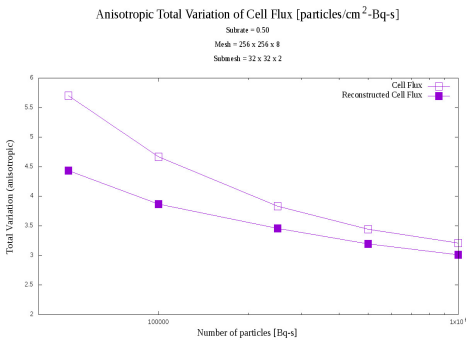


Figure 5.4: Total Variation (ℓ_{aniso}) of Mesh Tally for Cell Flux ($r_s = 0.5$)

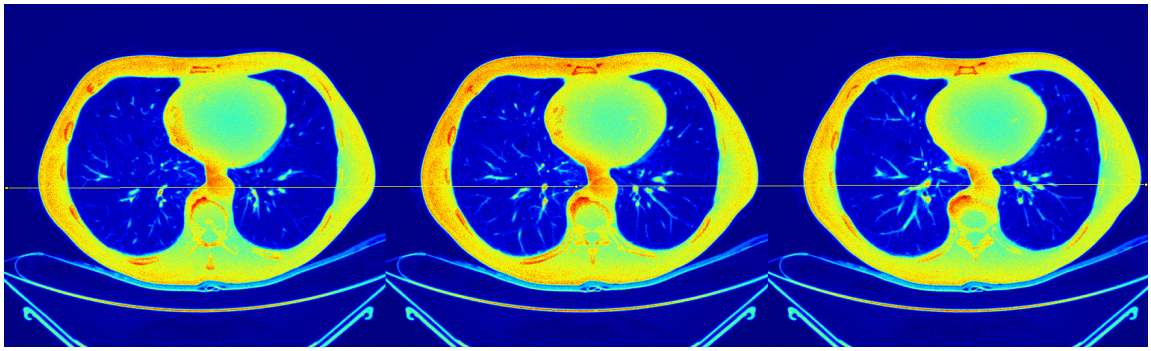


Figure 5.5: A heat map rendering of the number of photon collisions. Pixel value measurements are taken from horizontal yellow line super-imposed on the geometry for both the heat map rendered from standard tallies (Figure B.17 in Appendix B) and the heat map rendered from our reconstruction (Figure B.18 in Appendix B).

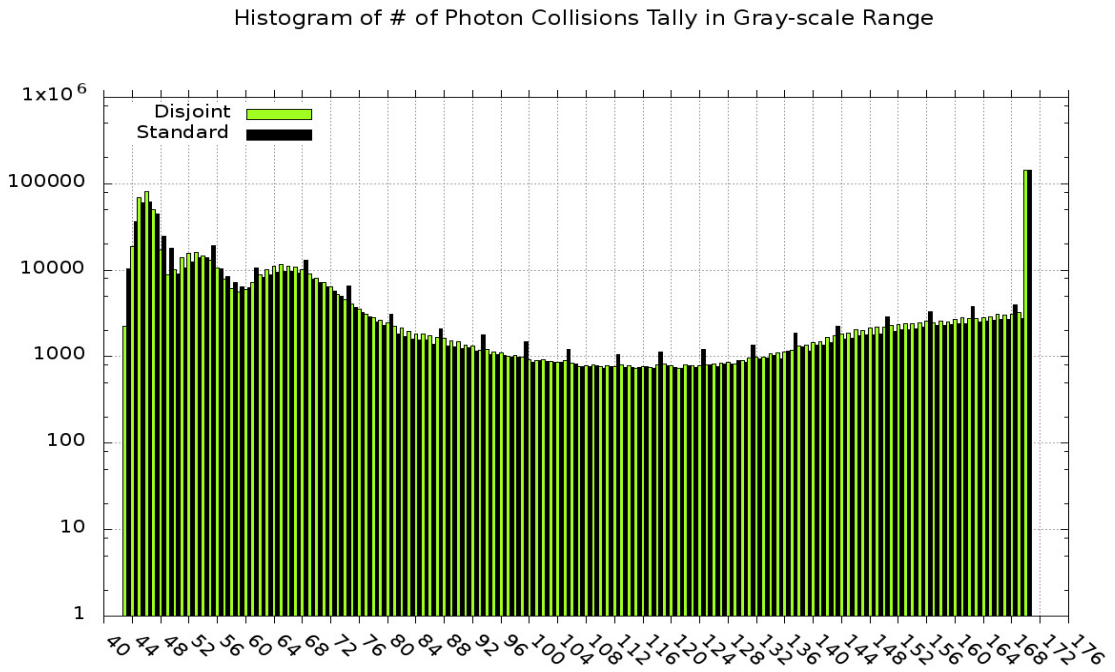


Figure 5.6: A histogram of the number of photon collisions tally mapped onto a gray-scale pixel range from [0, 256). This histogram is taken from the pixels intersecting with the horizontal line imposed on the image rendering of the tally shown in Figure 5.5.

5.2 Peak-Preserving Statistical Denoising

Evidence of the peak-preservation and statistical denoising is demonstrated in Figure 5.6. This DICOM simulation was produced from 1×10^8 incident photons and only photon collisions were recorded — an acceptable approximation to attenuation.

Analysis of the image shows that the localized peaks in the DICOM geometry are preserved in our method and, although a significant number of source photons were simulated, the standard tallies still contains small statistical fluctuations. These small statistical fluctuations result in a larger total variation and our methodology recognizes these fluctuation without smoothing out statistically-significant peaks in the data.

5.3 Solution Acceleration

5.3.1 Reactor Solution Acceleration

Analysis of Figures 5.1 and 5.2 suggest that at 250,000 particles the disjoint tally method scored with a subrate of 0.2 has a total variation approximately equivalent to a standard tally system at 1,000,000 particles and a disjoint tally with a subrate of 0.5 has a total variation approximately equivalent to a standard tally method at 500,000 particles. However, this alone cannot be taken as evidence of acceleration towards the solution. Instead, a more accurate test would be calculate the total variation between the solution produced at a series of intermediate particle counts compared to a highly resolved calculation.

In Figures 5.7, 5.8, 5.9, and 5.10, calculations of the total variation of the difference between the Monte Carlo solution at 2,500,000 and various intermediate numbers of particles have been compiled. Analysis of these figures indicate that the disjoint tally

scoring method does indeed provide superior results with fewer numbers of particles and the rate of acceleration in significantly affected by the subrate parameter. However, it should also be noted that although lower subrate parameters do provide better acceleration, the total variation difference appears to begin to approach an asymptote as the resolution improves and begins to provide slightly inferior results at 1,000,000 particles.

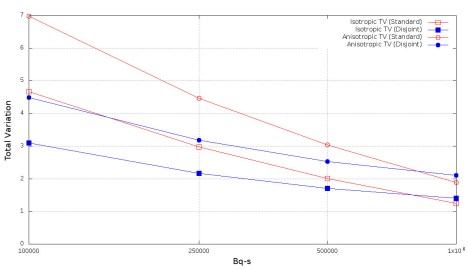


Figure 5.7: Total Variation of $\sum x_i$ for Energy Deposit Tally vs. Future Solution ($r_s = 0.2$)

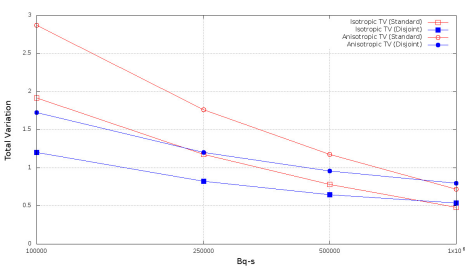


Figure 5.8: Total Variation of $\sum x_i$ for Cell Flux Tally vs. Future Solution ($r_s = 0.2$)

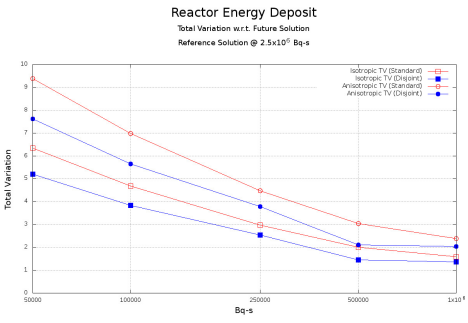


Figure 5.9: Total Variation of $\sum x_i$ for Energy Deposit Tally vs. Future Solution ($r_s = 0.5$)

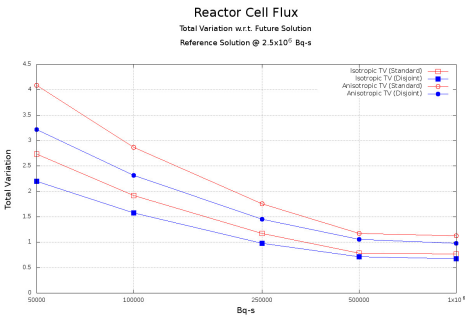


Figure 5.10: Total Variation of $\sum x_i$ for Cell Flux Tally vs. Future Solution ($r_s = 0.5$)

5.3.2 Shielding Solution Acceleration

An important Monte Carlo problem requiring a global solution is the shielding problem. This problem is computationally-intensive to simulate due to the limited number of particles that reach the furthest part of the mesh in an analog calculation. Variance reduction methods such as weight windows and importance sampling are commonly used to reduce the simulation time by spending more computational time in the simulation regions that analog calculations infrequently reach. We are presenting results for the shielding problem detailed in Section 3.4.2 that demonstrate that our method of solution acceleration through statistical denoising can be utilized in combination with variance reduction techniques. In these results, all simulations had an identical starting random number seed.

Figures 5.12 and 5.11 display the dose and cell flux results for a shielding simulation (Section 3.4.2) in a $80 \times 80 \times 80$ mesh reconstructed from $40 \times 40 \times 1$ blocks. The three-dimensional results are displayed in a two-dimensional format where contiguous slices in the Z-dimension are displayed vertically in sets of nine slices and then continued horizontally, as seen in Table 5.1.

Table 5.1: Z-slice layout of shielding results

0	10	...	70
1	11	...	71
...	...	...	79
8	17	...	80
9	18	...	

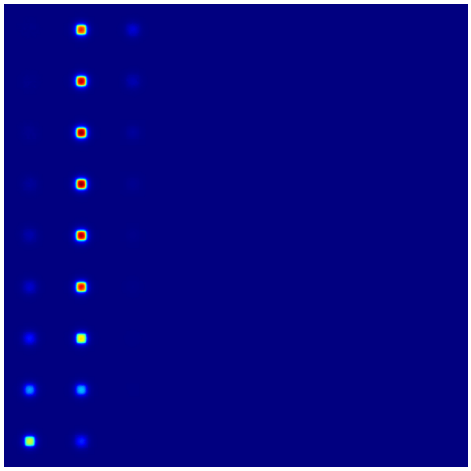


Figure 5.11: Shielding Cell Flux Tally

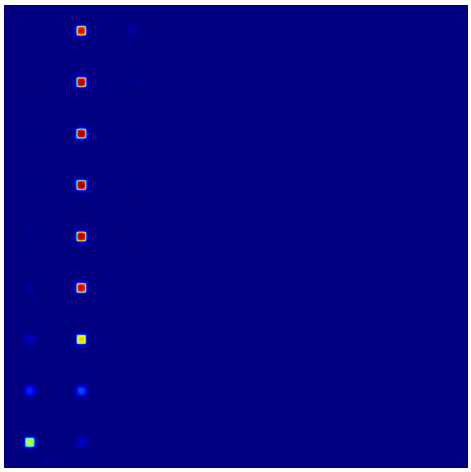


Figure 5.12: Shielding Dose Deposit Tally

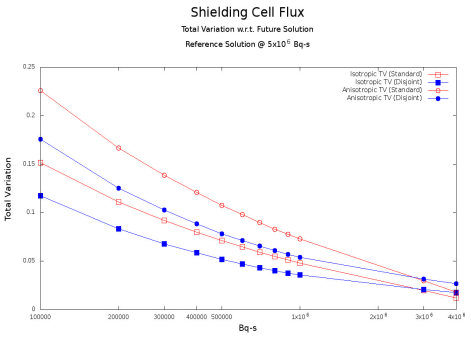


Figure 5.13: Total Variation of $\sum x_i$ for Cell Flux Tally vs. Future Solution ($r_s = 0.5$)

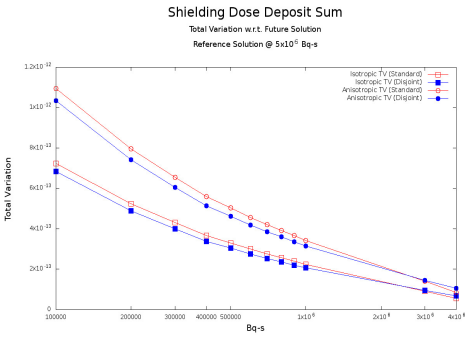


Figure 5.14: Total Variation of $\sum x_i$ for Dose Deposit Tally vs. Future Solution ($r_s = 0.5$)

5.4 Variance Reconstruction

Figures 5.15 and 5.16 display the comparison of the standard tally and disjoint tally total variation for the $\sum x_i^2$ (sum-squared) quantity. The combination of these values with the N_i (number of estimates) quantities permit the calculation of the variance, coefficient of

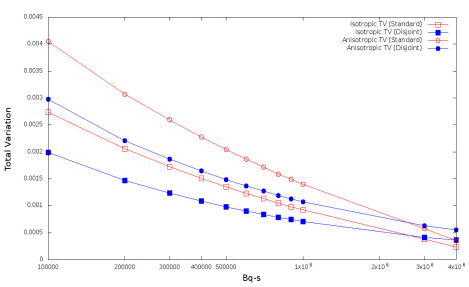


Figure 5.15: Total Variation of $\sum x_i^2$ for Cell Flux Tally vs. Future Solution ($r_s = 0.5$)

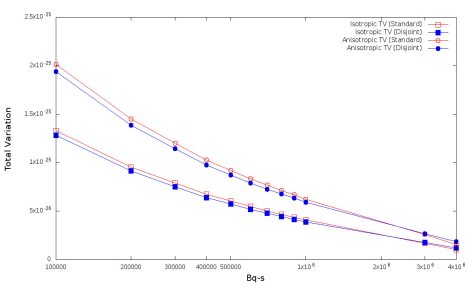


Figure 5.16: Total Variation of $\sum x_i^2$ for Dose Deposit Tally vs. Future Solution ($r_s = 0.5$)

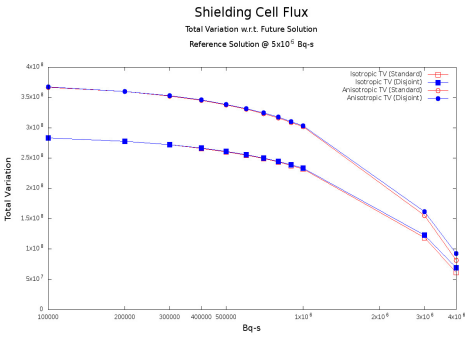


Figure 5.17: Total Variation of N_i (# of estimates) for Cell Flux Tally vs. Future Solution ($r_s = 0.5$)

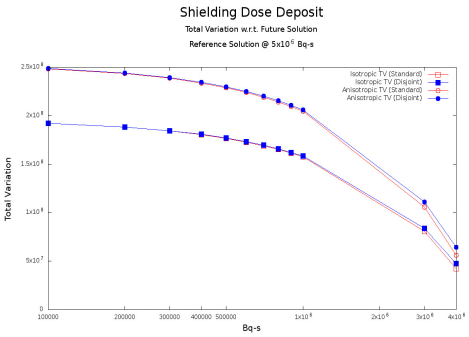


Figure 5.18: Total Variation of N_i (# of estimates) for Dose Deposit Tally vs. Future Solution ($r_s = 0.5$)

variation, RMS, and relative error. Inspection of Figures 5.15 and 5.16 with respect to Figures 5.13 and 5.14 will identify a similar slope in the approach towards the future solution and a similar relative difference between the disjoint and standard tallies at the intermediate number of particles. This, in combination with the nearly identical total variation of N_i with respect to the future solution for both the standard and disjoint tallies allows us to posit that the approximations of the statistical checks are accurate to the degree that the difference in these statistical quantities effectively demonstrate the level of

denoising our methodology provides. Analysis of Figures 5.17 and 5.18 demonstrate a high degree of precision in the reconstruction of the number of estimates for the tallies.

A comparison of the histograms in Figure 5.71 and Figure 5.73 demonstrate most of the same properties seen in Figure 5.5 with regards to peak-preservation in the statistical denoising process. Further comparison will reveal that Figure 5.73 has the same minimum, maximum, and mode as Figure 5.71 but with a modified mean and decreased standard deviation.

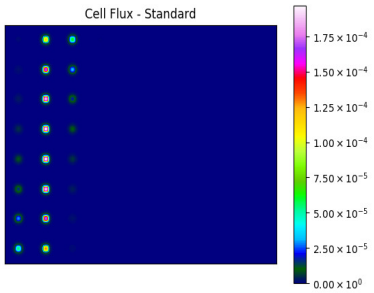


Figure 5.19: $\sum x_i$ for Cell Flux Standard Tally

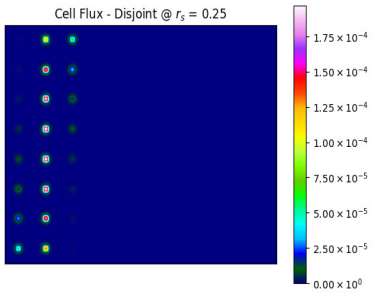


Figure 5.20: $\sum x_i$ for Cell Flux Disjoint Tally ($r_s = 0.25$)

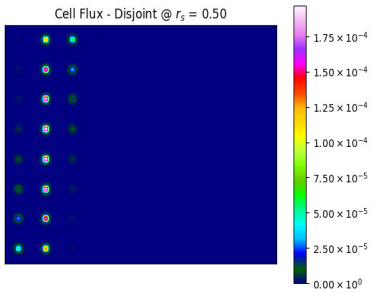


Figure 5.21: $\sum x_i$ for Cell Flux Disjoint Tally ($r_s = 0.50$)

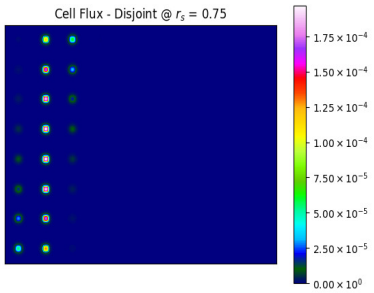


Figure 5.22: $\sum x_i$ for Cell Flux Disjoint Tally ($r_s = 0.75$)

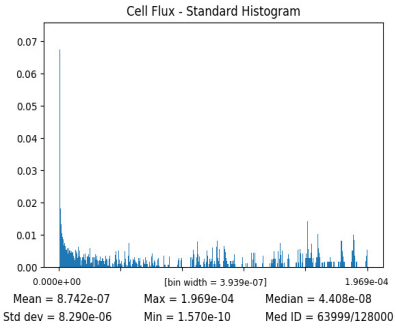


Figure 5.23: $\sum x_i$ Histogram for Figure 5.19 (standard)

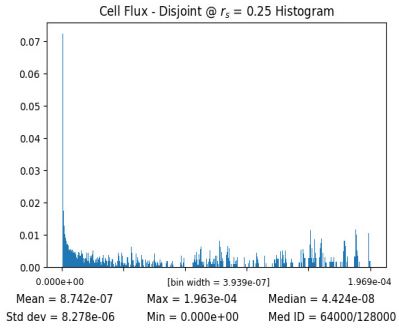


Figure 5.24: $\sum x_i$ Histogram for Figure 5.20 ($r_s = 0.25$)

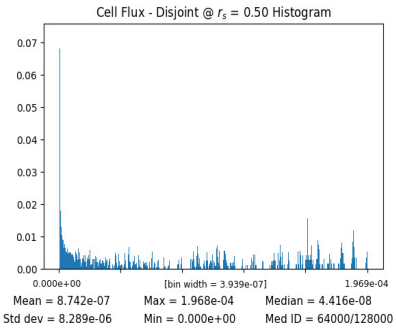


Figure 5.25: $\sum x_i$ Histogram for Figure 5.21 ($r_s = 0.50$)

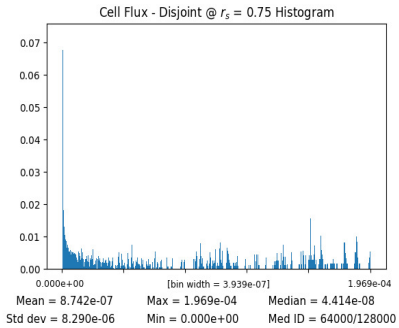


Figure 5.26: $\sum x_i$ Histogram for Figure 5.22 ($r_s = 0.75$)

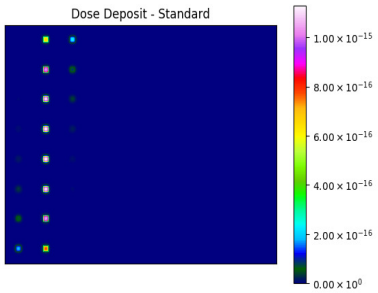


Figure 5.27: $\sum x_i$ for Dose Deposit Standard Tally

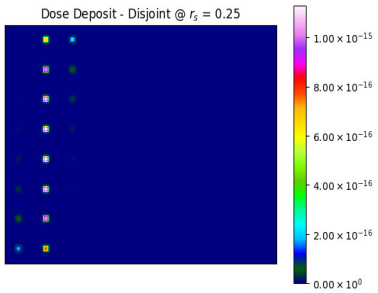


Figure 5.28: $\sum x_i$ for Dose Deposit Disjoint Tally ($r_s = 0.25$)

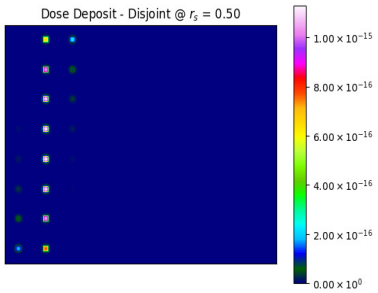


Figure 5.29: $\sum x_i$ for Dose Deposit Disjoint Tally ($r_s = 0.50$)

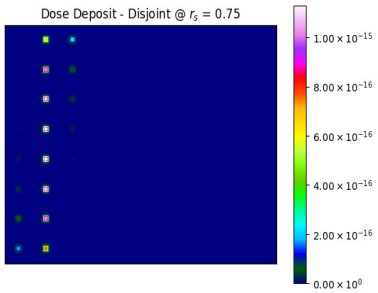


Figure 5.30: $\sum x_i$ for Dose Deposit Disjoint Tally ($r_s = 0.75$)

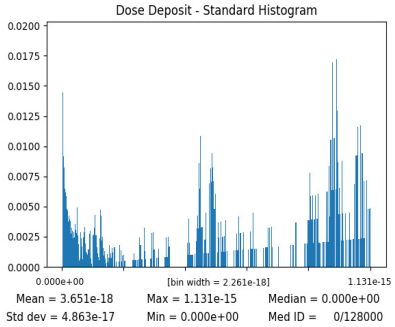


Figure 5.31: $\sum x_i$ Histogram for Figure 5.27 (standard)

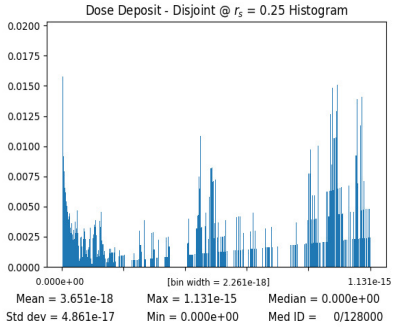


Figure 5.32: $\sum x_i$ Histogram for Figure 5.28 ($r_s = 0.25$)

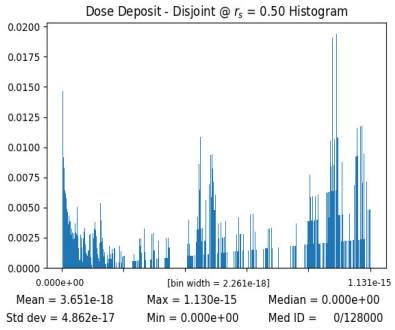


Figure 5.33: $\sum x_i$ Histogram for Figure 5.29 ($r_s = 0.50$)

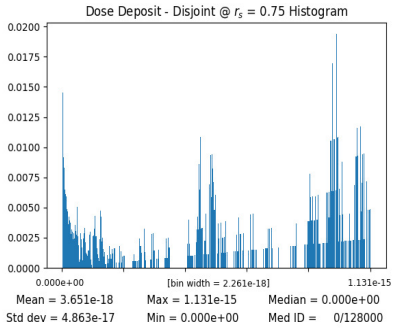


Figure 5.34: $\sum x_i$ Histogram for Figure 5.30 ($r_s = 0.75$)

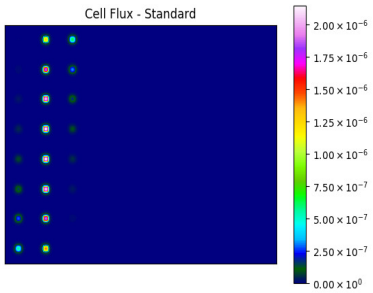


Figure 5.35: $\sum x_i^2$ for Cell Flux Standard Tally

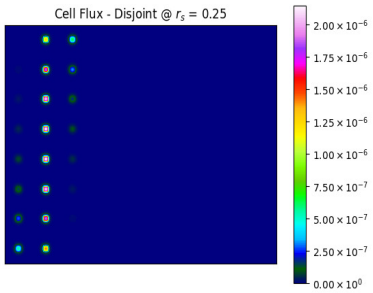


Figure 5.36: $\sum x_i^2$ for Cell Flux Disjoint Tally ($r_s = 0.25$)

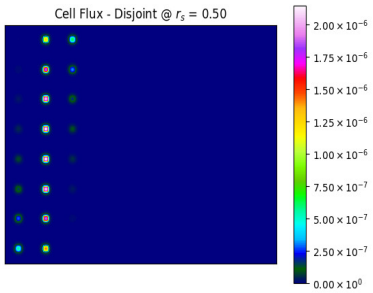


Figure 5.37: $\sum x_i^2$ for Cell Flux Disjoint Tally ($r_s = 0.50$)

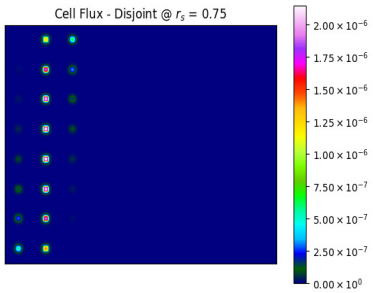


Figure 5.38: $\sum x_i^2$ for Cell Flux Disjoint Tally ($r_s = 0.75$)

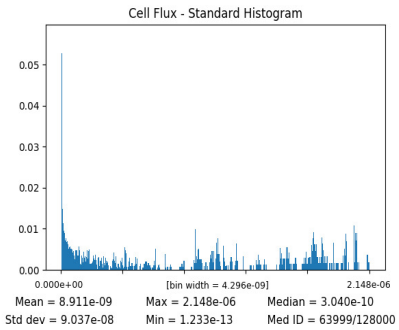


Figure 5.39: $\sum x_i^2$ Histogram for Figure 5.35 (standard)

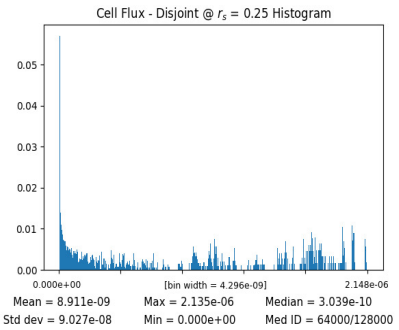


Figure 5.40: $\sum x_i^2$ Histogram for Figure 5.36 ($r_s = 0.25$)

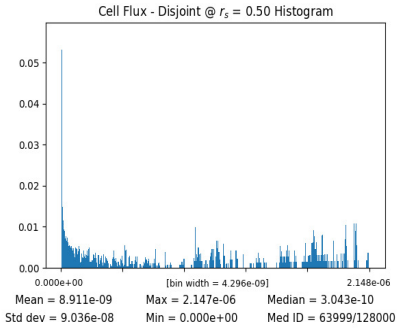


Figure 5.41: $\sum x_i^2$ Histogram for Figure 5.37 ($r_s = 0.50$)

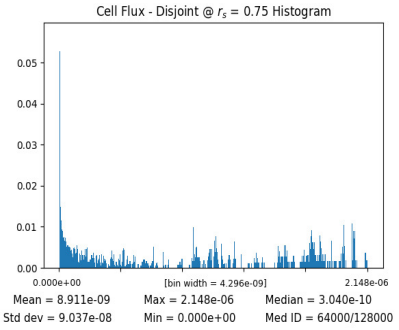


Figure 5.42: $\sum x_i^2$ Histogram for Figure 5.38 ($r_s = 0.75$)

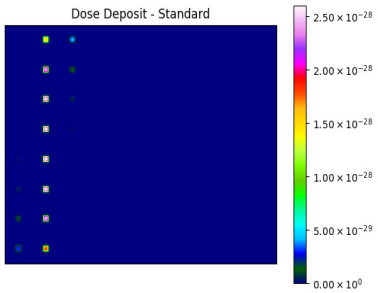


Figure 5.43: $\sum x_i^2$ for Dose Deposit Standard Tally

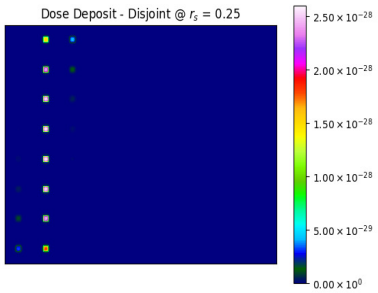


Figure 5.44: $\sum x_i^2$ for Dose Deposit Disjoint Tally ($r_s = 0.25$)

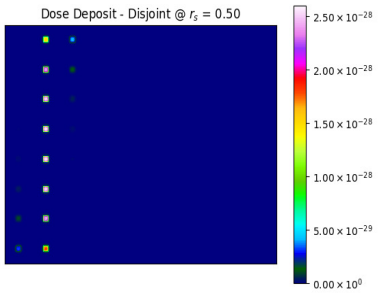


Figure 5.45: $\sum x_i^2$ for Dose Deposit Disjoint Tally ($r_s = 0.50$)

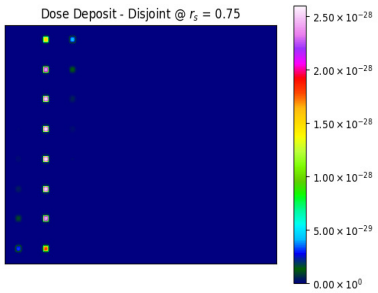


Figure 5.46: $\sum x_i^2$ for Dose Deposit Disjoint Tally ($r_s = 0.75$)

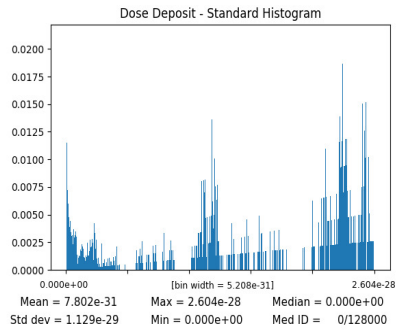


Figure 5.47: $\sum x_i^2$ Histogram for Figure 5.43 (standard)

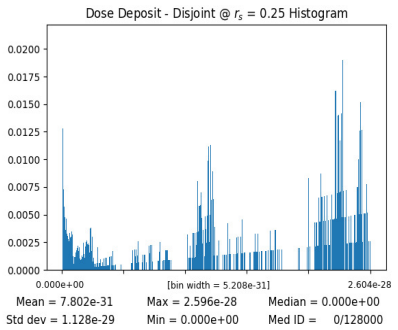


Figure 5.48: $\sum x_i^2$ Histogram for Figure 5.44 ($r_s = 0.25$)

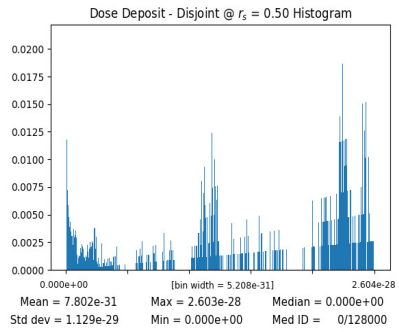


Figure 5.49: $\sum x_i^2$ Histogram for Figure 5.45 ($r_s = 0.50$)

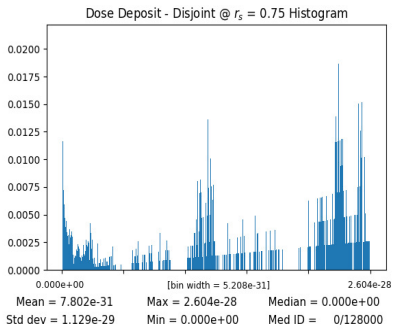


Figure 5.50: $\sum x_i^2$ Histogram for Figure 5.46 ($r_s = 0.75$)

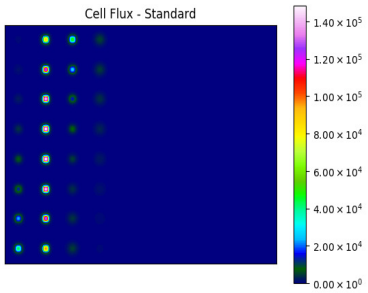


Figure 5.51: Number of Estimates for Cell Flux Standard Tally

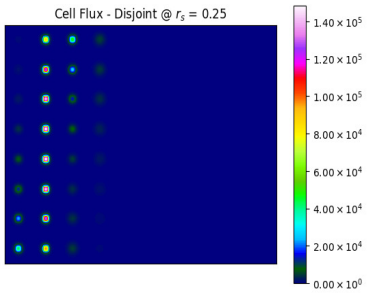


Figure 5.52: Number of Estimates for Cell Flux Disjoint Tally ($r_s = 0.25$)

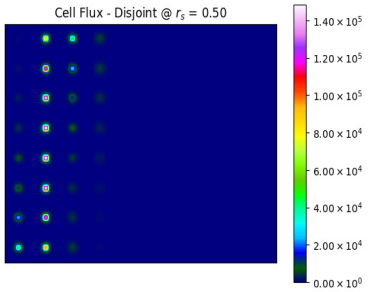


Figure 5.53: Number of Estimates for Cell Flux Disjoint Tally ($r_s = 0.50$)

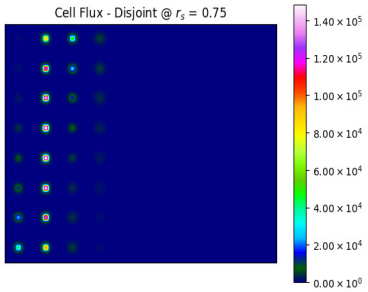


Figure 5.54: Number of Estimates for Cell Flux Disjoint Tally ($r_s = 0.75$)

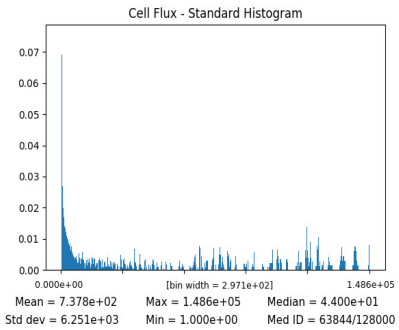


Figure 5.55: Number of Estimates Histogram for Figure 5.51 (standard)

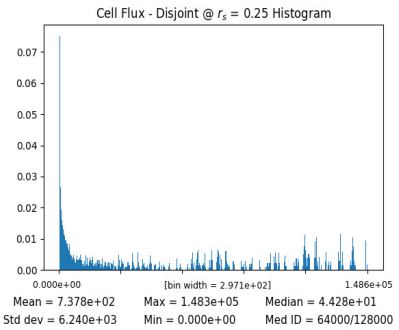


Figure 5.56: Number of Estimates Histogram for Figure 5.52 ($r_s = 0.25$)

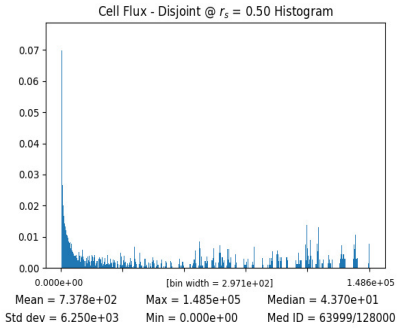


Figure 5.57: Number of Estimates Histogram for Figure 5.53 ($r_s = 0.50$)

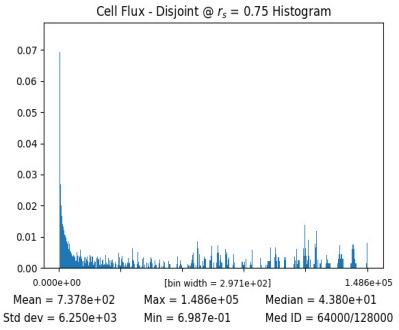


Figure 5.58: Number of Estimates Histogram for Figure 5.54 ($r_s = 0.75$)

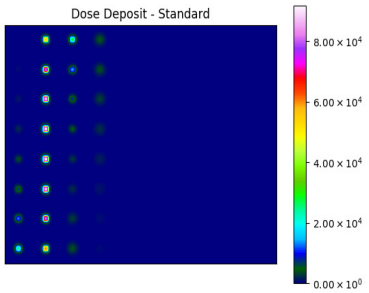


Figure 5.59: Number of Estimates for Dose Deposit Standard Tally

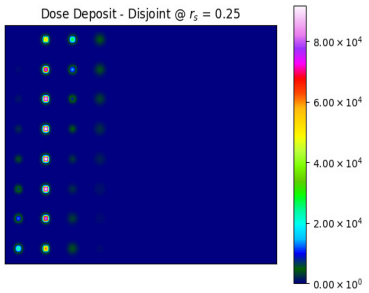


Figure 5.60: Number of Estimates for Dose Deposit Disjoint Tally ($r_s = 0.25$)

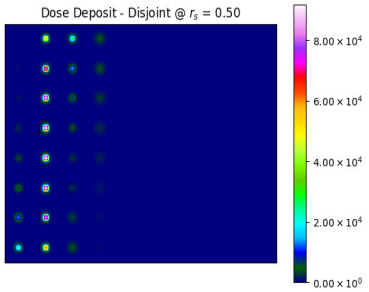


Figure 5.61: Number of Estimates for Dose Deposit Disjoint Tally ($r_s = 0.50$)

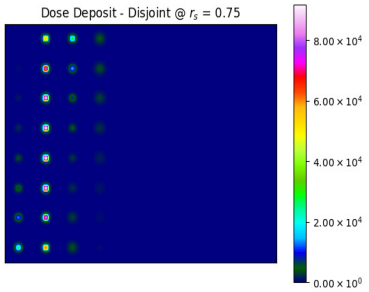


Figure 5.62: Number of Estimates for Dose Deposit Disjoint Tally ($r_s = 0.75$)

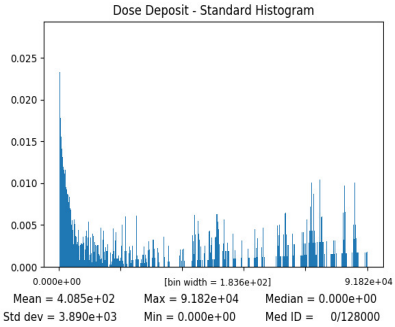


Figure 5.63: Number of Estimates Histogram for Figure 5.59 (standard)

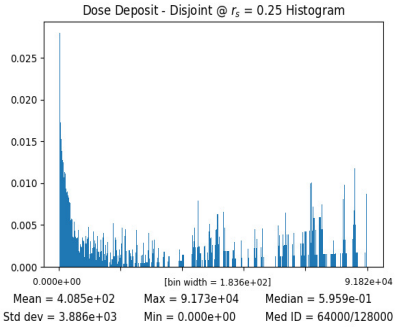


Figure 5.64: Number of Estimates Histogram for Figure 5.60 ($r_s = 0.25$)

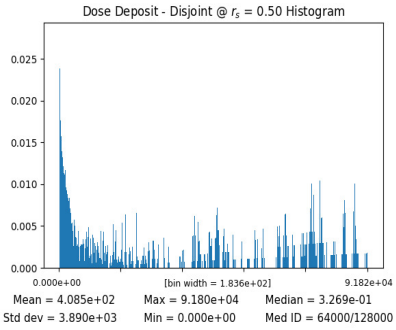


Figure 5.65: Number of Estimates Histogram for Figure 5.61 ($r_s = 0.50$)

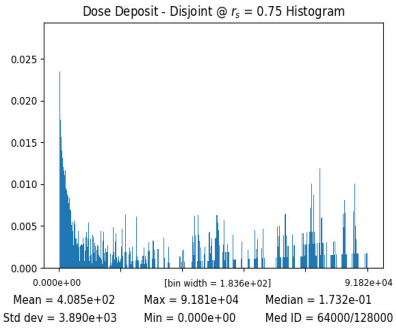


Figure 5.66: Number of Estimates Histogram for Figure 5.62 ($r_s = 0.75$)

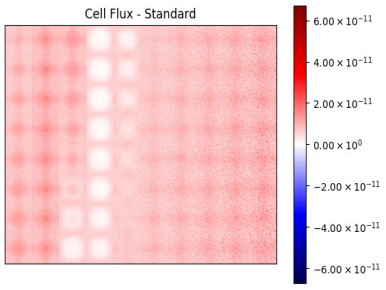


Figure 5.67: Variance for Cell Flux Standard Tally

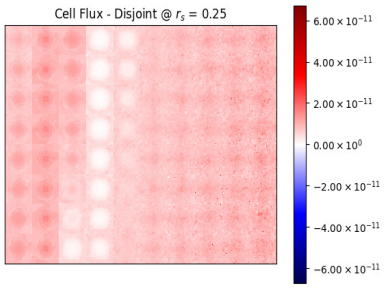


Figure 5.68: Variance for Cell Flux Disjoint Tally ($r_s = 0.25$)

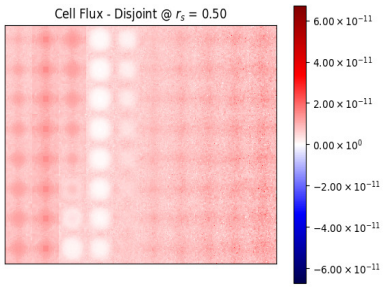


Figure 5.69: Variance for Cell Flux Disjoint Tally ($r_s = 0.50$)

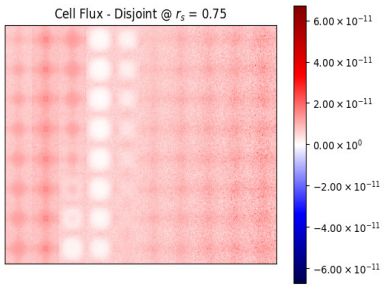


Figure 5.70: Variance for Cell Flux Disjoint Tally ($r_s = 0.75$)

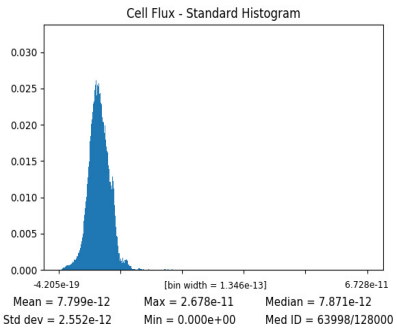


Figure 5.71: Variance Histogram for Figure 5.67 (standard)

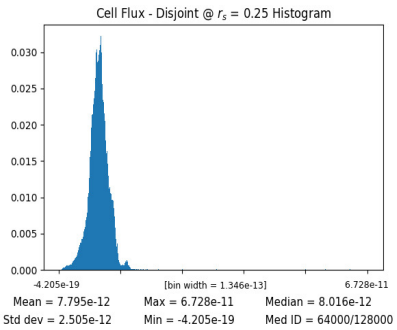


Figure 5.72: Variance Histogram for Figure 5.68 ($r_s = 0.25$)

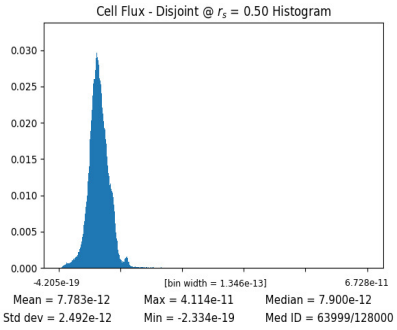


Figure 5.73: Variance Histogram for Figure 5.69 ($r_s = 0.50$)

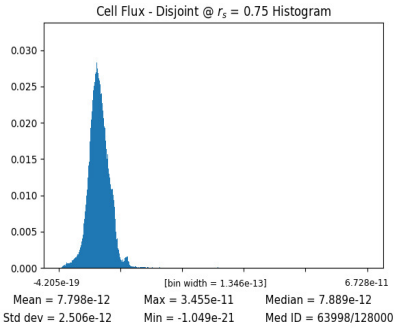


Figure 5.74: Variance Histogram for Figure 5.70 ($r_s = 0.75$)

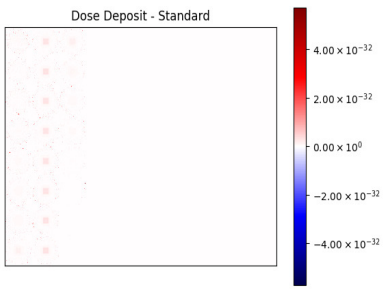


Figure 5.75: Variance for Dose Deposit Standard Tally

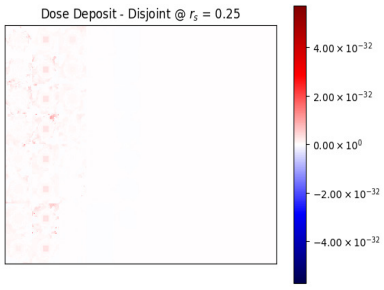


Figure 5.76: Variance for Dose Deposit Disjoint Tally ($r_s = 0.25$)

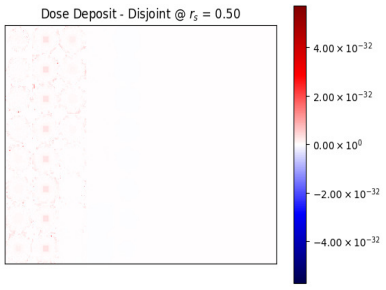


Figure 5.77: Variance for Dose Deposit Disjoint Tally ($r_s = 0.50$)

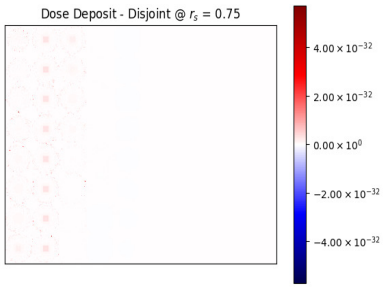


Figure 5.78: Variance for Dose Deposit Disjoint Tally ($r_s = 0.75$)

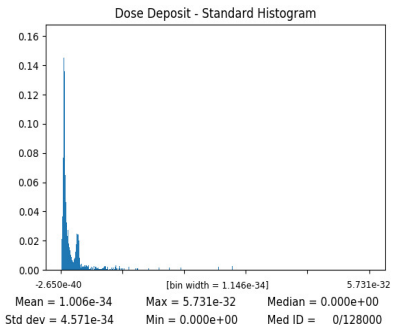


Figure 5.79: Variance Histogram for Figure 5.75 (standard)

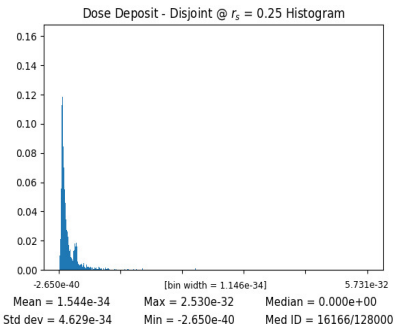


Figure 5.80: Variance Histogram for Figure 5.76 ($r_s = 0.25$)

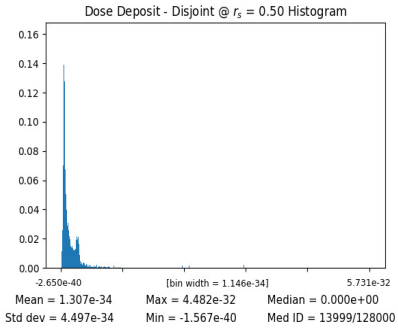


Figure 5.81: Variance Histogram for Figure 5.77 ($r_s = 0.50$)

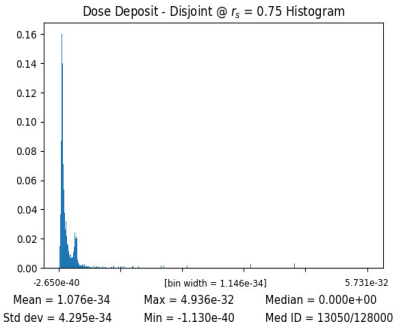


Figure 5.82: Variance Histogram for Figure 5.78 ($r_s = 0.75$)

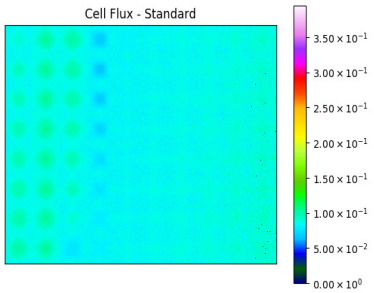


Figure 5.83: RMS for Cell Flux Standard Tally

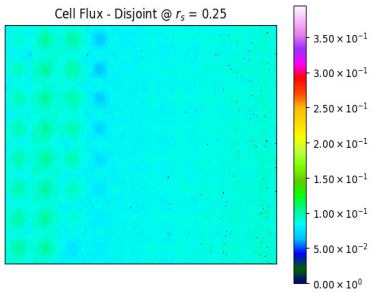


Figure 5.84: RMS for Cell Flux Disjoint Tally ($r_s = 0.25$)

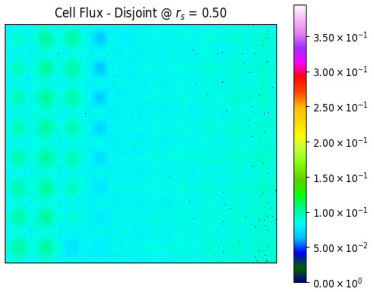


Figure 5.85: RMS for Cell Flux Disjoint Tally ($r_s = 0.50$)

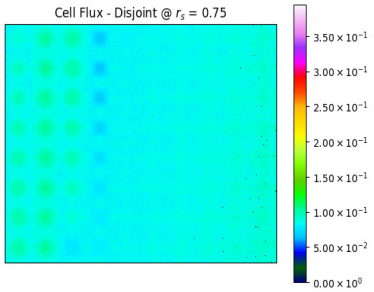


Figure 5.86: RMS for Cell Flux Disjoint Tally ($r_s = 0.75$)

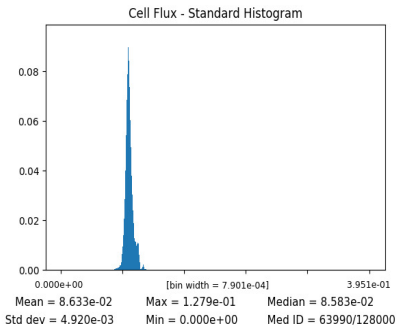


Figure 5.87: RMS Histogram for Figure 5.83 (standard)

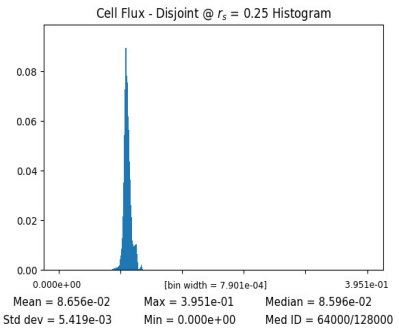


Figure 5.88: RMS Histogram for Figure 5.84 ($r_s = 0.25$)

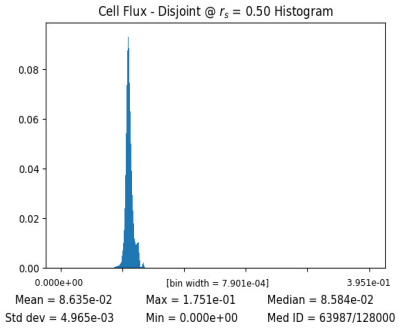


Figure 5.89: RMS Histogram for Figure 5.85 ($r_s = 0.50$)

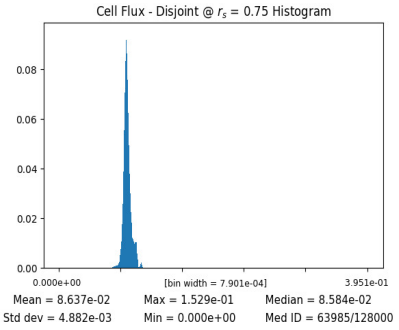


Figure 5.90: RMS Histogram for Figure 5.86 ($r_s = 0.75$)

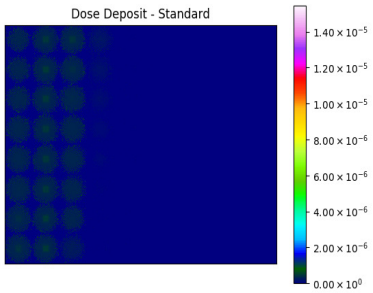


Figure 5.91: RMS for Dose Deposit Standard Tally

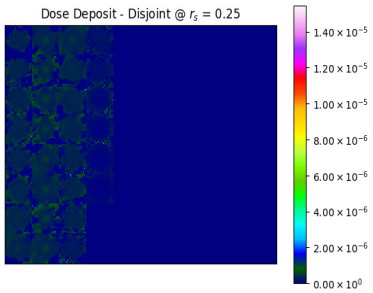


Figure 5.92: RMS for Dose Deposit Disjoint Tally ($r_s = 0.25$)

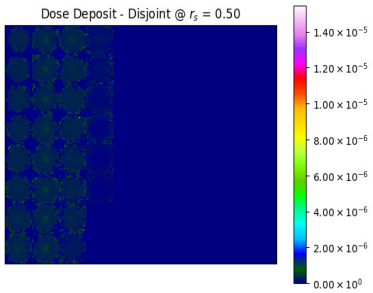


Figure 5.93: RMS for Dose Deposit Disjoint Tally ($r_s = 0.50$)

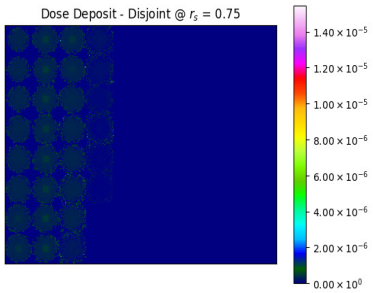


Figure 5.94: RMS for Dose Deposit Disjoint Tally ($r_s = 0.75$)

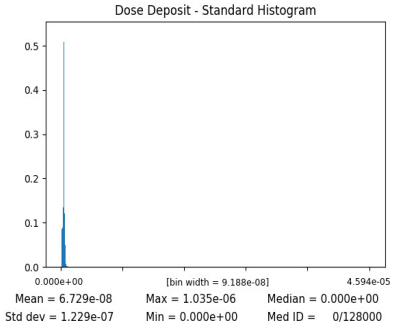


Figure 5.95: RMS Histogram for Figure 5.91 (standard)

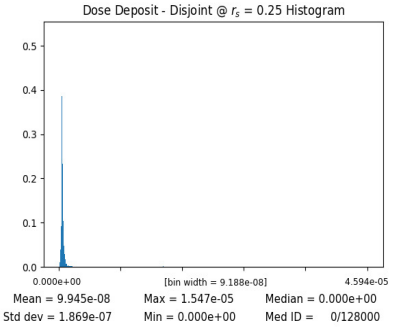


Figure 5.96: RMS Histogram for Figure 5.92 ($r_s = 0.25$)

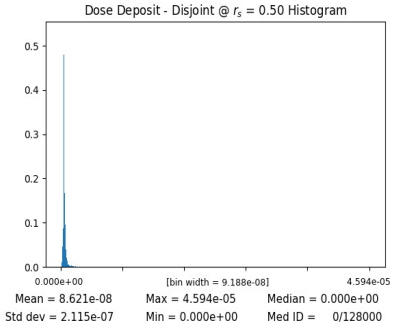


Figure 5.97: RMS Histogram for Figure 5.93 ($r_s = 0.50$)

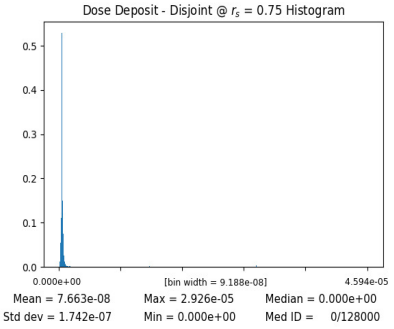


Figure 5.98: RMS Histogram for Figure 5.94 ($r_s = 0.75$)

5.5 Memory Reduction Results

Measurement of the runtime reduction of memory can be found in Figure 5.99, Figure 5.100, Figure 5.101, Figure 5.102, Figure 5.103, and Figure 5.104,

The formula for memory savings is calculated with:

$$\% \text{ Memory Reduction} = \left(1 - \frac{\text{disjoint}}{\text{standard}}\right) * 100$$

where disjoint is the total memory allocated by the program running with only disjoint scoring and standard is the total memory allocated by the program running with a standard scoring method.

Analysis of Figures 5.99-5.104 will show that the memory savings are affected most significantly by the number of tallies and and the subrate parameter. The number of threads has a non-negligible but less significant effect on the memory savings.

For consideration on subrate, Appendix C, has several reconstructions at subrates of of 0.25, 0.3, 0.5, 0.7, and 0.75. The memory reduction at a subrate value of 0.1 is included in the graph primarily for informational purposes. A reconstruction at a subrate of 0.1 will not produce a solution consistent with simulation, however on the other end of the spectrum, a reconstruction at the subrate of 0.9 will produce a nearly perfect reconstruction.

Memory reduction results in this section were compared a memory-efficient vector of pointers to the statistical analysis class detailed in Section 3.4 as a replacement to the less memory-efficient map[1] of pointers keyed by an integer, as is ubiquitously found in the

[1]When $> 40\%$ of voxel IDs are populated in the map, the memory allocation of the map exceed the memory allocation of a vector of size n_v, where n_v is the number of voxels

Geant4 examples.

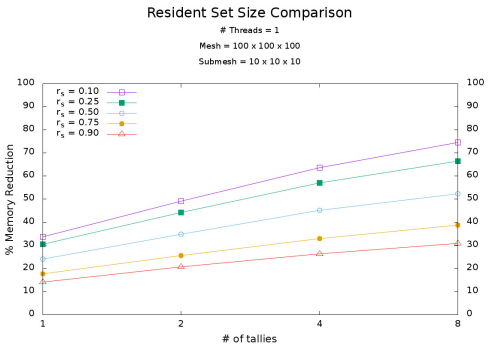


Figure 5.99: Memory Reduction Results for 1 thread — $100 \times 100 \times 100$ mesh in $10 \times 10 \times 10$ sub-blocks

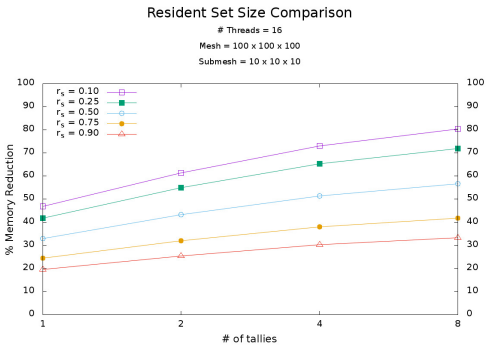


Figure 5.100: Memory Reduction Results for 16 threads — $100 \times 100 \times 100$ mesh in $10 \times 10 \times 10$ sub-blocks

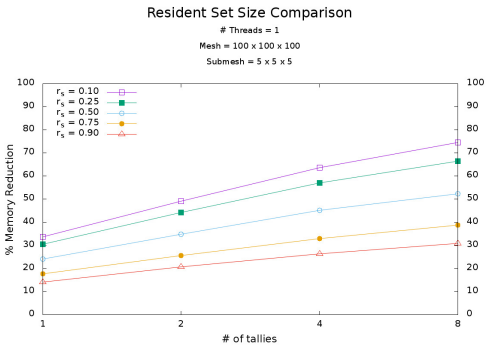


Figure 5.101: Memory Reduction Results for 1 thread — $100 \times 100 \times 100$ mesh in $5 \times 5 \times 5$ sub-blocks

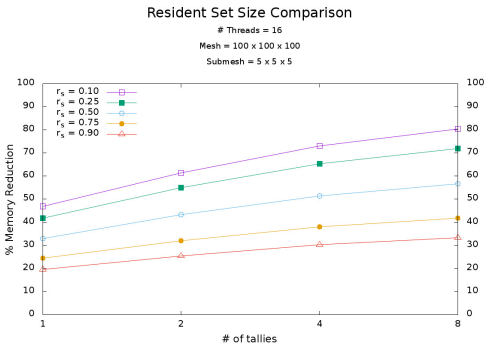


Figure 5.102: Memory Reduction Results for 16 threads — $100 \times 100 \times 100$ mesh in $5 \times 5 \times 5$ sub-blocks

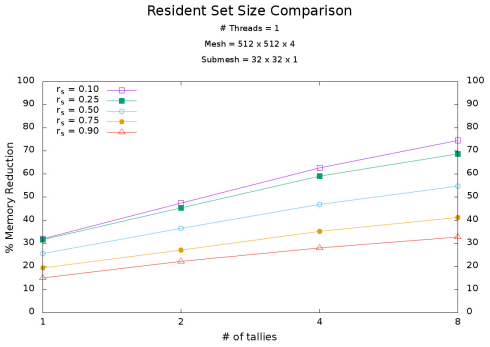


Figure 5.103: Memory Reduction Results for 1 thread — $512 \times 512 \times 4$ mesh in $32 \times 32 \times 1$ sub-blocks

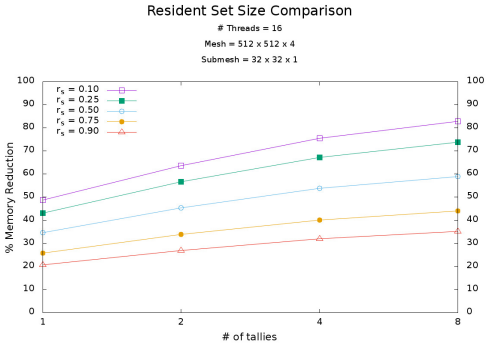


Figure 5.104: Memory Reduction Results for 16 threads — $512 \times 512 \times 4$ mesh in $32 \times 32 \times 1$ sub-blocks

5.6 Reconstruction Summary

1. Excellent memory reduction at 1 thread and 1 tally
2. Memory reduction is dependent on subrate, number of quantities scored, and number of threads
3. Balance of memory reduction with reconstruction quality and denoising at $r_s = 0.5$
4. Large memory reduction but lower reconstruction quality at $r_s \leq 0.3$
5. Reconstructions were precise with respect to the standard tallies and still reduce memory at $r_s \geq 0.7$
6. Many different tally types were successfully reconstructed
7. Reconstruction of variance proved successful to an acceptable approximation

6. CONCLUSION

In this dissertation we have presented a beneficial methodology for mesh-based scoring in Monte Carlo transport codes. Memory reduction through the use of disjoint tallies was demonstrated against a memory-efficient vector of standard tallies, instead of comparison against the standard, yet less memory-efficient, scoring tallies using maps in Geant4. Even against this memory-efficient implementation, the disjoint tallies were able to achieve around 25% memory reduction for 1 thread and 1 scoring tally at the standard subrate of 0.5. With lower subrates and 16 threads with 8 scoring tallies, memory reduction exceeded 80%.

This dissertation established a method for quantifying the divergence between a lower fidelity Monte Carlo calculation and a high-fidelity Monte Carlo calculation in Section 5.1. Utilizing this method, we were able to demonstrate that the disjoint tally system not only converges to the same value as standard tallies as the fidelity improves, but that at a lower fidelity, the statistical denoising in the disjoint tally method decreases the divergence between itself and the high-fidelity more rapidly than standard tallies.

Two options are available utilizing the disjoint tally system. The first option is to implement the disjoint tally system directly into the software package and provide the compressed results to the library for reconstruction. The benefits of this approach are a reduction in runtime memory of the Monte Carlo software and the statistical denoising, which has been shown to approach the solution of higher fidelity calculations faster than standard tallies. We have also shown that the methodology is capable of producing a good approximation to the variance, which permit the computation of a confidence interval. The methodology has been successfully tested against nearly all the standard

mesh-based tallies in Monte Carlo and provides additional information about statistical resolution through the presence of negative values in overly noisy regions. The accumulation of these memory-reducing and solution accelerating benefits outweigh the rare occasion where the runtime increase exceeds 5%. The second option is to continue using standard tallies but utilizing the statistical denoising properties of the method to post-process Monte Carlo simulations.

REFERENCES

[1] E. Candes and J. Romberg, "l1-magic: Recovery of Sparse Signals via Convex Programming." `http://users.ece.gatech.edu/justin/l1magic/downloads/l1magic.pdf`, 2005. Accessed: 2016-07-27.

[2] A. Heidari and D. Saeedkia, "A 2D camera design with a single-pixel detector," in *2009 34th International Conference on Infrared, Millimeter, and Terahertz Waves*, pp. 1–2, Sept 2009.

[3] P. Saidi, M. Sadeghi, and C. Tenreiro, "Variance Reduction of Monte Carlo Simulation in Nuclear Engineering Field," in *Theory and Applications of Monte Carlo Simulations* (Chan, Victor (Wai Kin), ed.), ch. 07, Rijeka: InTech, 2013.

[4] S. Foucart and H. Rauhut, *A Mathematical Introduction to Compressive Sensing*. Springer New York, 2013.

[5] E. Candes and T. Tao, "The Dantzig selector: Statistical estimation when p is much larger than n," *ArXiv Mathematics e-prints*, June 2005.

[6] G. B. Dantzig, A. Orden, and P. Wolfe, "The generalized simplex method for minimizing a linear form under linear inequality restraints," *Pacific Journal of Mathematics*, vol. 5, no. 2, pp. 183–195, 1955.

[7] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, July 1948.

[8] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 623–656, Oct 1948.

[9] D. L. Donoho and X. Huo, "Uncertainty principles and ideal atomic decomposition," *IEEE Trans. Inf. Theor.*, vol. 47, pp. 2845–2862, Sept. 2006.

[10] J. C. Wagner, D. E. Peplow, and S. W. Mosher, "FW-CADIS Method for Global and Regional Variance Reduction of Monte Carlo Radiation Transport Calculations," *Nuclear Science and Engineering*, vol. 176, no. 1, pp. 37–57, 2014.

[11] C. Sanderson and R. Curtin, "Armadillo: C++ Linear Algebra Library." `http://arma.sourceforge.net`, 2016. Accessed: 2016-07-27.

[12] Boost, "Boost C++ Libraries." `http://www.boost.org/`, 2017. Last accessed 2017-03-07.

[13] Kitware, Inc. and Contributors, "CMake - Cross Platform Makefile Generator." `https://cmake.org/`, 2017. Last accessed 2016-07-16.

[14] D. E. King, "Dlib-ml: A Machine Learning Toolkit," *Journal of Machine Learning Research*, vol. 10, pp. 1755–1758, 2009.

[15] D. M. Beazley, "SWIG: An Easy to Use Tool for Integrating Scripting Languages with C and C++," in *Proceedings of the 4th Conference on USENIX Tcl/Tk Workshop, 1996 - Volume 4*, TCLTK'96, (Berkeley, CA, USA), pp. 15–15, USENIX Association, 1996.

[16] T. Natschläger, "ArmaNpy." `https://sourceforge.net/p/armanpy/wiki/Home/`, 2014. Last accessed 2016-08-07.

[17] S. L. Xiaoye, “An Overview of SuperLU: Algorithms, Implementation, and User Interface,” *ACM Trans. Mathematical Software*, vol. 31, pp. 302–325, September 2005.

APPENDIX A

GEANT4 PARAMETERS

A.1 Reactor

- Physics Lists
  - G4EmStandardPhysics_option3
  - G4HadronPhysicQGSP_BIC (Inelastic)
    - QGSP - Quark Gluon String (fragmentation) + Precompound (de-excitation)
    - BIC - Binary Cascade for inelastic scattering
  - G4HadronElasticPhysics
  - G4DecayPhysics
  - G4IonPhysicsPHP
  - G4StepLimiterPhysics
  - G4RadioactiveDecayPhysics
- Tracking cut: 0.1 mm
- Results for reactor bundle-type geometry

  - 24 bundles (5×5 array minus 1 bundle)
  - 145 pins/bundle arranged in hexagon
  - Each fuel pin is G4_URANIUM_DIOXIDE (NIST) and Zr cladding
  - One bundle of control rods with 80% Ag, 15% In, and 5% Cd composition
- Moderating material is water
- Random particles
  - e^- [100 keV — 10 MeV]
  - neutrons [10 eV — 10 MeV]
  - gamma [1 keV — 1 MeV]
- Random position and momentum direction
  - Randomly selected fuel pin
  - Random location within fuel pin
  - Random energy within specified range
  - Isotropically emitted

A.2 Shielding

- Physics Lists
  - G4EmStandardPhysics_option4 (Penelope)

  - G4EmExtraPhysics (Synchrotron Radiation and Gamma-Nuclear)
  - G4StoppingPhysics
  - G4IonQMDPhysics
  - G4IonElasticPhysics
  - G4HadronPhysicQGSP_BERT (Inelastic)
    * QGSP - Quark Gluon String (fragmentation) + Precompound (de-excitation)
    * BERT - Bertini Cascade for inelastic scattering
  - G4HadronElasticPhysics
  - G4DecayPhysics
  - G4IonPhysicsPHP
  - G4StepLimiterPhysics
  - G4RadioactiveDecayPhysics
- Tracking cut: 1.0 cm
- Incident neutron source
  - Incident neutron energy [1 eV — 14 MeV]
- Random position and momentum direction
  - Random location within fuel region

  - Energy from Watt Spectrum (neutron induced)
  - Isotropically emitted

A.3 DICOM

- Physics Lists
  - G4EmStandardPhysics_option3 (Livermore)
  - G4EmExtraPhysics (Synchrotron Radiation and Gamma-Nuclear)
  - G4StoppingPhysics
  - G4IonQMDPhysics
  - G4IonElasticPhysics
  - G4HadronPhysicQGSP_BERT (Inelastic)
    - QGSP - Quark Gluon String (fragmentation) + Precompound (de-excitation)
    - BERT - Bertini Cascade for inelastic scattering
  - G4HadronElasticPhysics
  - G4DecayPhysics
  - G4IonPhysicsPHP
  - G4StepLimiterPhysics

  - G4RadioactiveDecayPhysics

- Tracking cut: 0.1 mm

- X-ray source

  - gamma[1] [90 keV — 120 keV]

- Random position on imaginary cylinder surrounding patient phantom

- Momentum direction emitted in 45 degree spread towards center of patient in XY-plane

[1] No “x-ray” in Geant4

APPENDIX B

RECONSTRUCTION RESULTS

B.1 Solution Acceleration

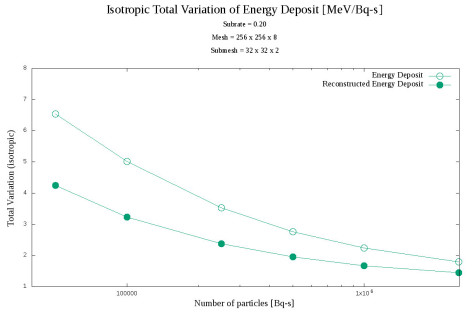


Figure B.1: Total Variation (ℓ_1) of Mesh Tally for Energy Deposit ($r_s = 0.2$)

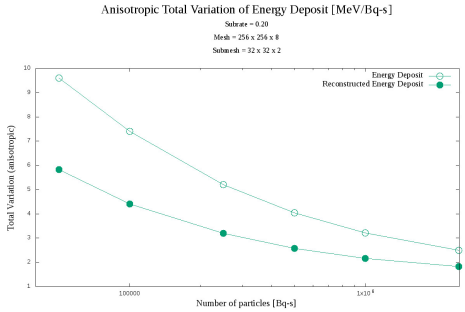


Figure B.2: Total Variation (ℓ_∞) of Mesh Tally for Energy Deposit ($r_s = 0.2$)

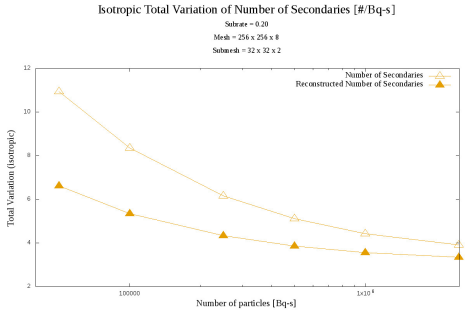


Figure B.3: Total Variation (ℓ_1) of Mesh Tally for Number of Secondaries ($r_s = 0.2$)

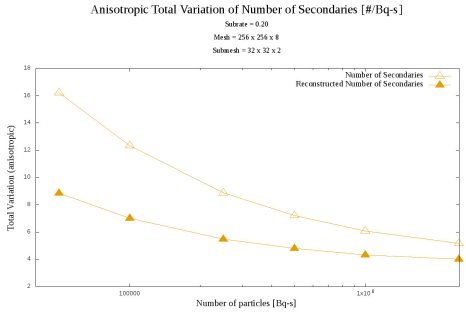


Figure B.4: Total Variation (ℓ_∞) of Mesh Tally for Number of Secondaries ($r_s = 0.2$)

B.2 DICOM

The DICOM tally heatmaps in this appendix section were generated from linear distribution of 90-120 keV source photons after 1×10^8 particles. Geometry rendered from three contiguous DICOM CT slices detailed in Section 3.16. The disjoint and standard tallies scored onto a $512 \times 512 \times 3$ mesh and the disjoint tallies were reconstructed in $16 \times 16 \times 3$ sub-blocks using a subrate of $r_s = 0.5$.

B.2.1 Core Reconstruction Quantities for Statistical Reconstruction

B.2.1.1 Energy Deposit

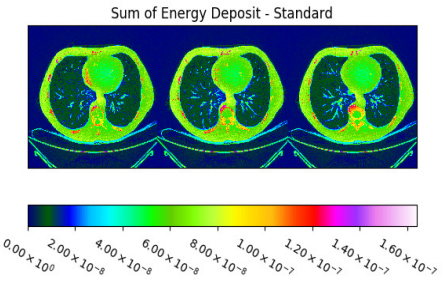


Figure B.5: Sum for Energy Deposit Tally $\left(\sum x_i\right)$ — Standard

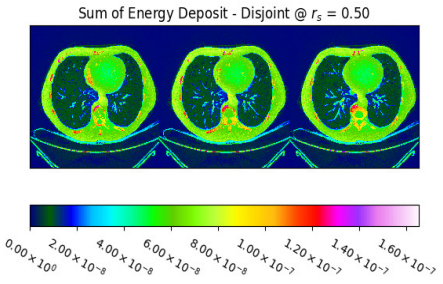


Figure B.6: Sum for Energy Deposit Tally $\left(\sum x_i\right)$ — Disjoint

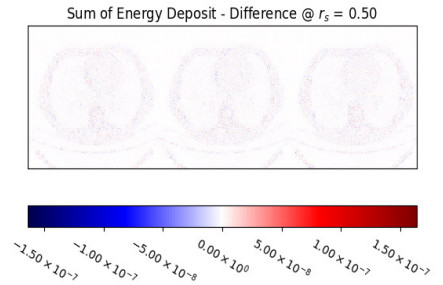


Figure B.7: Absolute Difference in Sum for Energy Deposit Tally $\left(\sum x_i\right)$ between Figure B.6 and Figure B.5 (using min/max of tally)

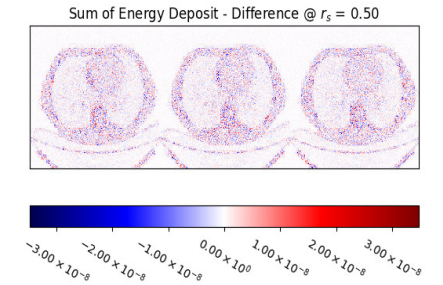


Figure B.8: Absolute Difference in Sum for Energy Deposit Tally $\left(\sum x_i\right)$ between Figure B.6 and Figure B.5 (using min/max of data)

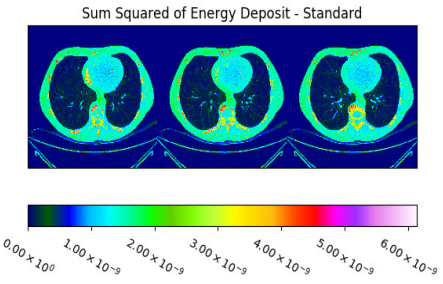


Figure B.9: Sum Squared for Energy Deposit Tally $\left(\sum x_i^2\right)$ — Standard

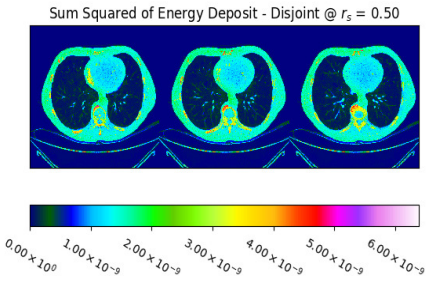


Figure B.10: Sum Squared for Energy Deposit Tally $\left(\sum x_i^2\right)$ — Disjoint

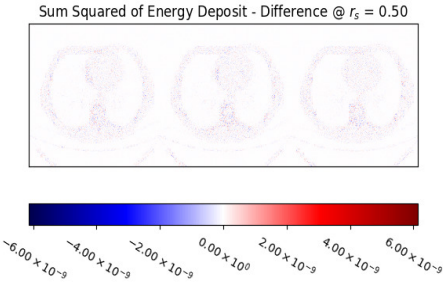


Figure B.11: Absolute Difference in Sum Squared for Energy Deposit Tally $\left(\sum x_i^2\right)$ between Figure B.10 and Figure B.9 (using min/max of tally)

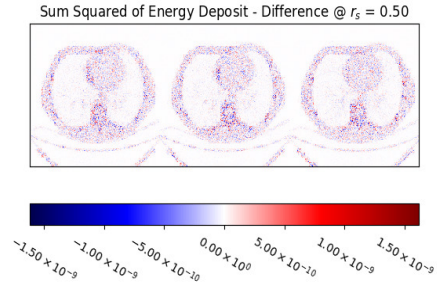


Figure B.12: Absolute Difference in Sum Squared for Energy Deposit Tally $\left(\sum x_i^2\right)$ between Figure B.10 and Figure B.9 (using min/max of data)

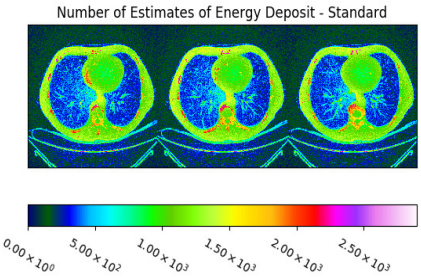


Figure B.13: Number of Estimates for Energy Deposit Tally $\left(\sum n_i\right)$ — Standard

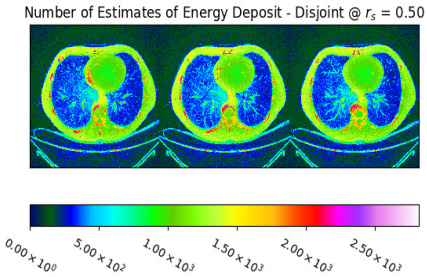


Figure B.14: Number of Estimates for Energy Deposit Tally $\left(\sum n_i\right)$ — Disjoint

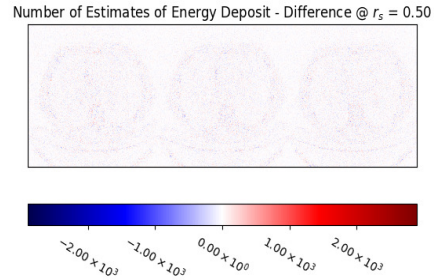


Figure B.15: Absolute Difference in Number of Estimates for Energy Deposit Tally $\left(\sum n_i\right)$ between Figure B.14 and Figure B.13 (using min/max of tally)

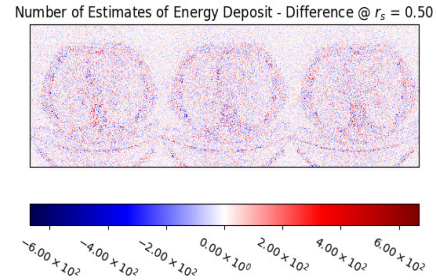


Figure B.16: Absolute Difference in Number of Estimates for Energy Deposit Tally $\left(\sum n_i\right)$ between Figure B.14 and Figure B.13 (using min/max of data)

B.2.1.2 Number of Collisions

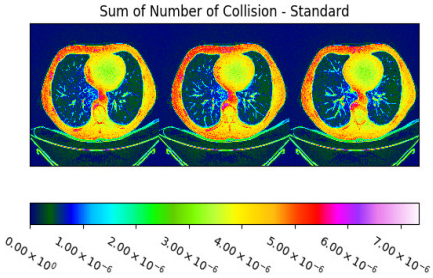


Figure B.17: Sum for Number of Collisions Tally $\left(\sum x_i\right)$ — Standard

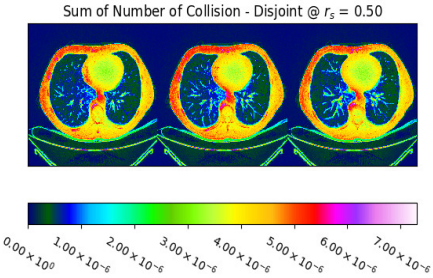


Figure B.18: Sum for Number of Collisions Tally $\left(\sum x_i\right)$ — Disjoint

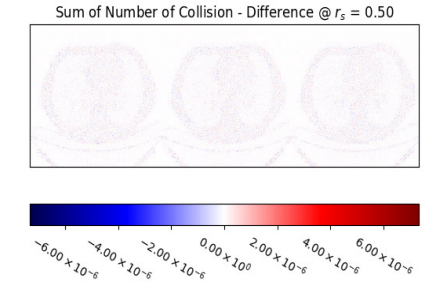


Figure B.19: Absolute Difference in Sum for Number of Collisions Tally $\left(\sum x_i\right)$ between Figure B.18 and Figure B.17 (using min/max of tally)

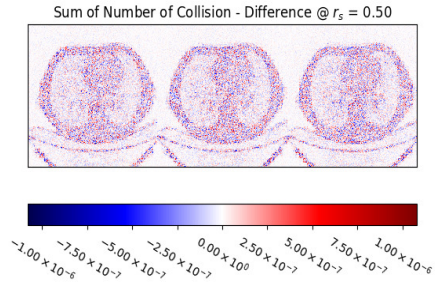


Figure B.20: Absolute Difference in Sum for Number of Collisions Tally $\left(\sum x_i\right)$ between Figure B.18 and Figure B.17 (using min/max of data)

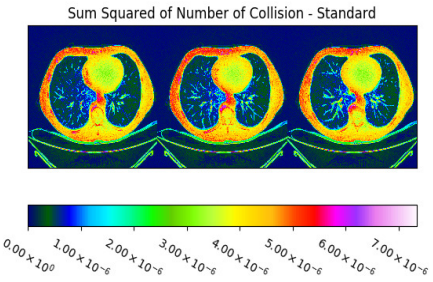


Figure B.21: Sum Squared for Number of Collisions Tally $\left(\sum x_i^2\right)$ — Standard

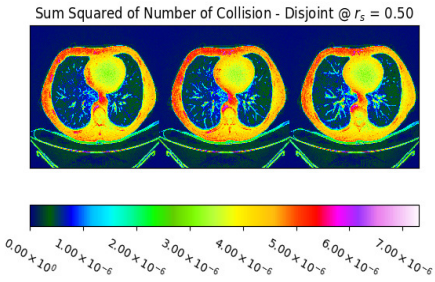


Figure B.22: Sum Squared for Number of Collisions Tally $\left(\sum x_i^2\right)$ — Disjoint

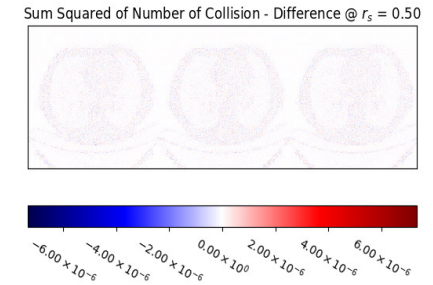


Figure B.23: Absolute Difference in Sum Squared for Number of Collisions Tally $\left(\sum x_i^2\right)$ between Figure B.22 and Figure B.21 (using min/max of tally)

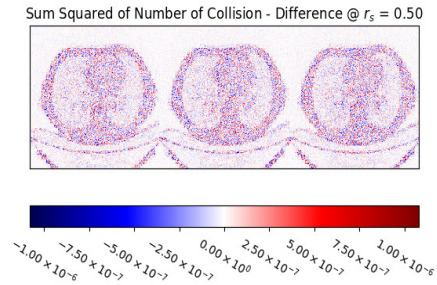


Figure B.24: Absolute Difference in Sum Squared for Number of Collisions Tally $\left(\sum x_i^2\right)$ between Figure B.22 and Figure B.21 (using min/max of data)

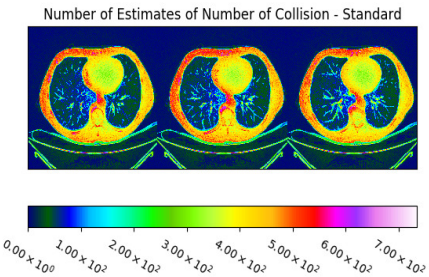


Figure B.25: Number of Estimates for Number of Collisions Tally $\left(\sum n_i\right)$ — Standard

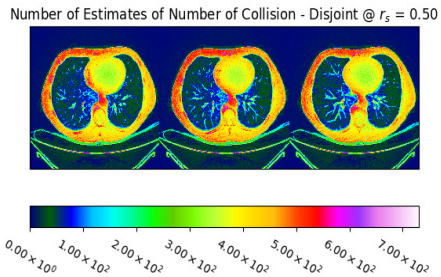


Figure B.26: Number of Estimates for Number of Collisions Tally $\left(\sum n_i\right)$ — Disjoint

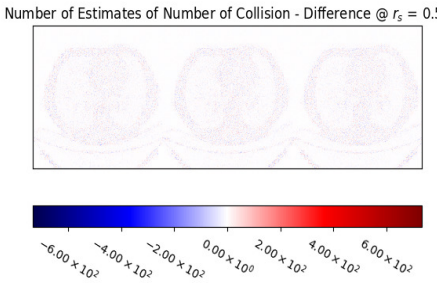


Figure B.27: Absolute Difference in Number of Estimates for Number of Collisions Tally $\left(\sum n_i\right)$ between Figure B.26 and Figure B.25 (using min/max of tally)

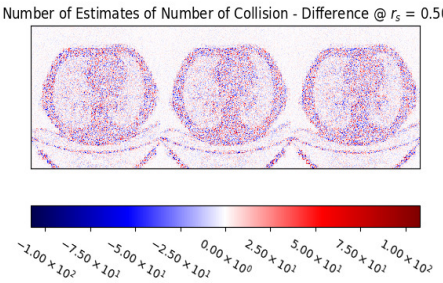


Figure B.28: Absolute Difference in Number of Estimates for Number of Collisions Tally $\left(\sum n_i\right)$ between Figure B.26 and Figure B.25 (using min/max of data)

B.2.1.3 Termination

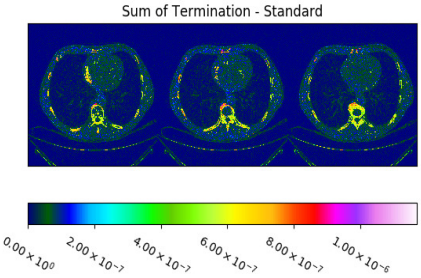


Figure B.29: Sum for Termination Tally $\left(\sum x_i\right)$ — Standard

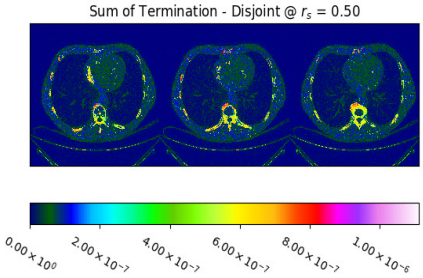


Figure B.30: Sum for Termination Tally $\left(\sum x_i\right)$ — Disjoint

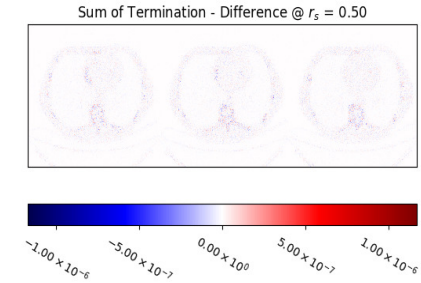


Figure B.31: Absolute Difference in Sum for Termination Tally $\left(\sum x_i\right)$ between Figure B.30 and Figure B.29 (using min/max of tally)

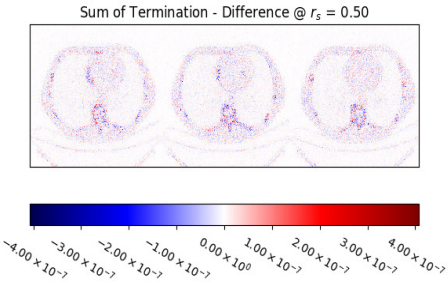


Figure B.32: Absolute Difference in Sum for Termination Tally $\left(\sum x_i\right)$ between Figure B.30 and Figure B.29 (using min/max of data)

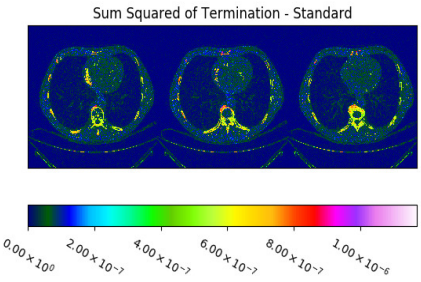


Figure B.33: Sum Squared for Termination Tally $\left(\sum x_i^2\right)$ — Standard

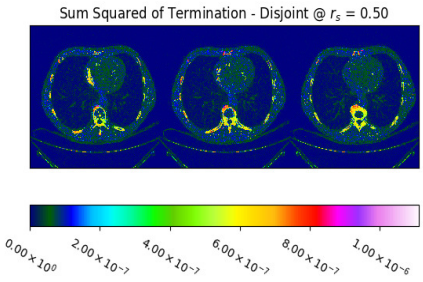


Figure B.34: Sum Squared for Termination Tally $\left(\sum x_i^2\right)$ — Disjoint

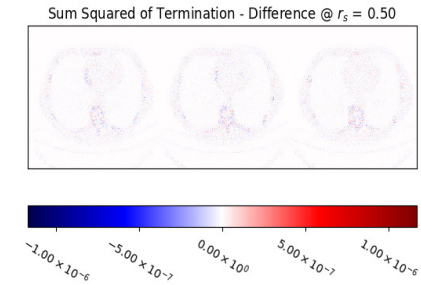


Figure B.35: Absolute Difference in Sum Squared for Termination Tally $\left(\sum x_i^2\right)$ between Figure B.34 and Figure B.33 (using min/max of tally)

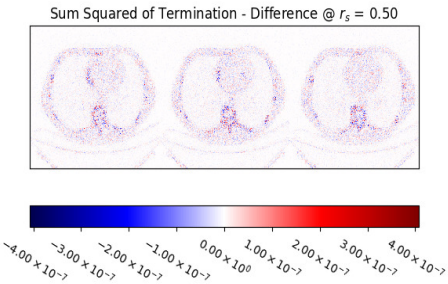


Figure B.36: Absolute Difference in Sum Squared for Termination Tally $\left(\sum x_i^2\right)$ between Figure B.34 and Figure B.33 (using min/max of data)

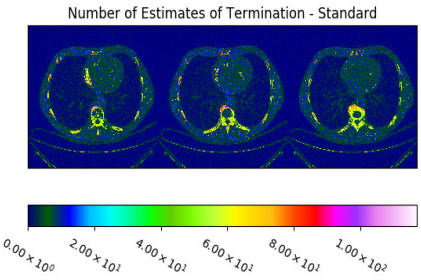


Figure B.37: Number of Estimates for Termination Tally $\left(\sum n_i\right)$ — Standard

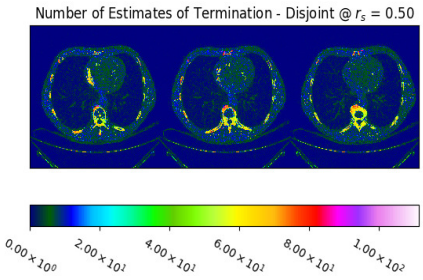


Figure B.38: Number of Estimates for Termination Tally $\left(\sum n_i\right)$ — Disjoint

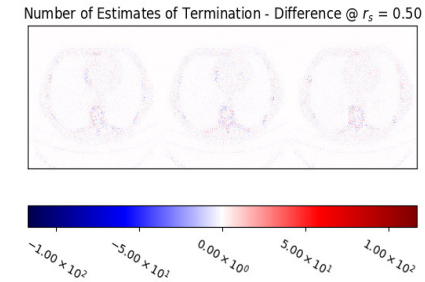


Figure B.39: Absolute Difference in Number of Estimates for Termination Tally $\left(\sum n_i\right)$ between Figure B.38 and Figure B.37 (using min/max of tally)

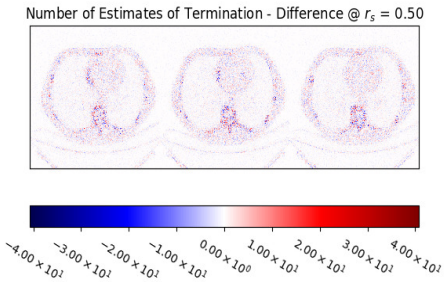


Figure B.40: Absolute Difference in Number of Estimates for Termination Tally $\left(\sum n_i\right)$ between Figure B.38 and Figure B.37 (using min/max of data)

B.2.2 Statistical Reconstructions

B.2.2.1 Energy Deposit

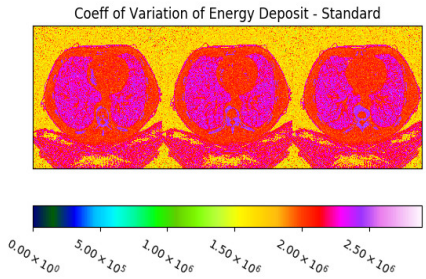


Figure B.41: Coefficient of Variation for Energy Deposit Tally — Standard

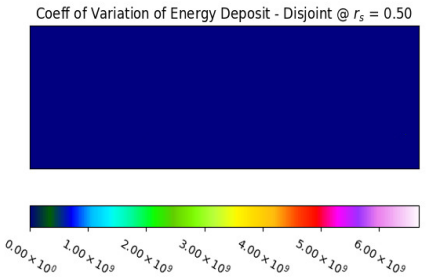


Figure B.42: Coefficient of Variation for Energy Deposit Tally — Disjoint

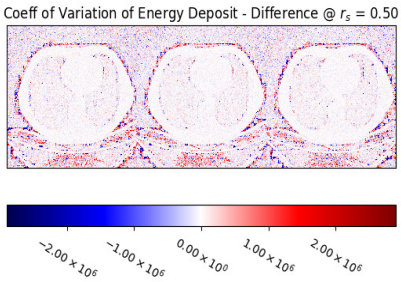


Figure B.43: Absolute Difference in Coefficient of Variation for Energy Deposit Tally between Figure B.42 and Figure B.41 (using min/max of tally)

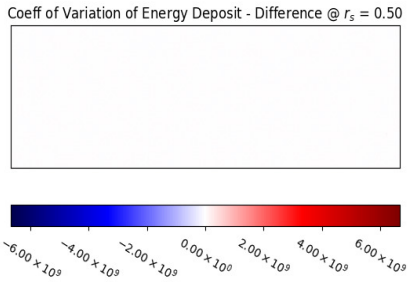


Figure B.44: Absolute Difference in Coefficient of Variation for Energy Deposit Tally between Figure B.42 and Figure B.41 (using min/max of data)

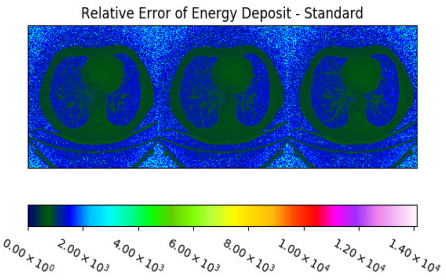


Figure B.45: Relative Error for Energy Deposit Tally — Standard

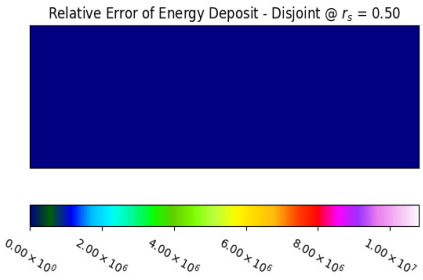


Figure B.46: Relative Error for Energy Deposit Tally — Disjoint

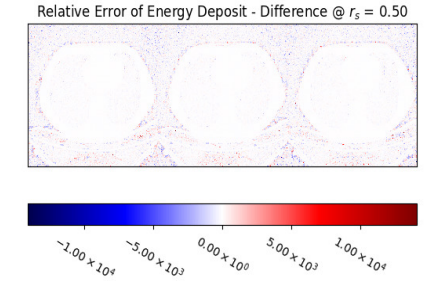


Figure B.47: Absolute Difference in Relative Error for Energy Deposit Tally between Figure B.46 and Figure B.45 (using min/max of tally)

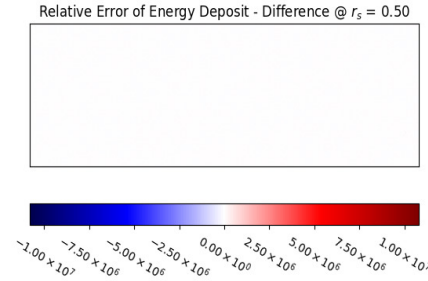


Figure B.48: Absolute Difference in Relative Error for Energy Deposit Tally between Figure B.46 and Figure B.45 (using min/max of data)

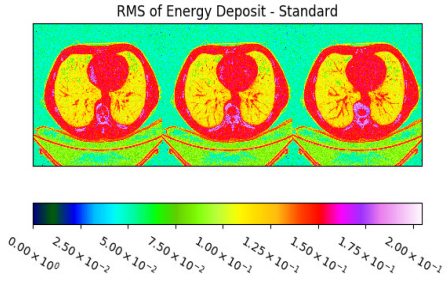


Figure B.49: Root Mean Squared (RMS) for Energy Deposit Tally — Standard

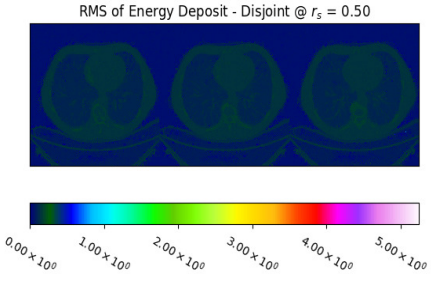


Figure B.50: Root Mean Squared (RMS) for Energy Deposit Tally — Disjoint

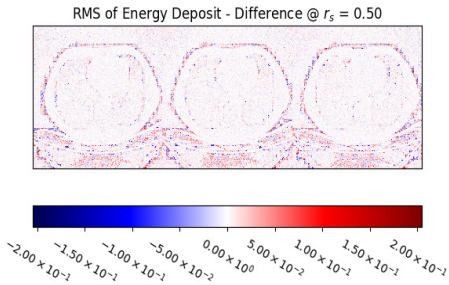


Figure B.51: Absolute Difference in Root Mean Squared (RMS) for Energy Deposit Tally between Figure B.50 and Figure B.49 (using min/max of tally)

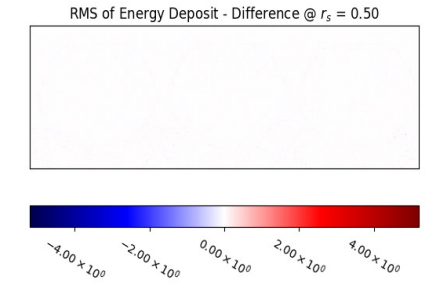


Figure B.52: Absolute Difference in Root Mean Squared (RMS) for Energy Deposit Tally between Figure B.50 and Figure B.49 (using min/max of data)

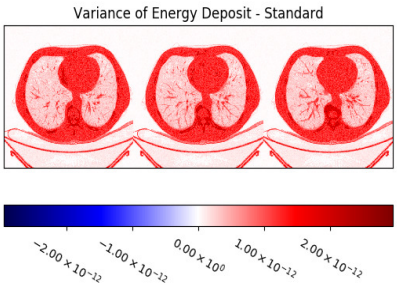


Figure B.53: Variance for Energy Deposit Tally — Standard

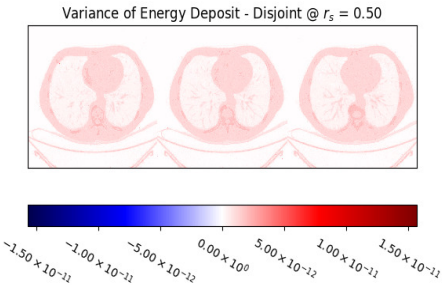


Figure B.54: Variance for Energy Deposit Tally — Disjoint

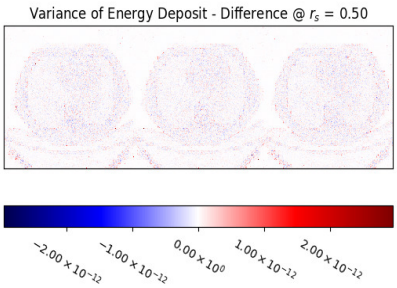


Figure B.55: Absolute Difference in Variance for Energy Deposit Tally between Figure B.54 and Figure B.53 (using min/max of tally)

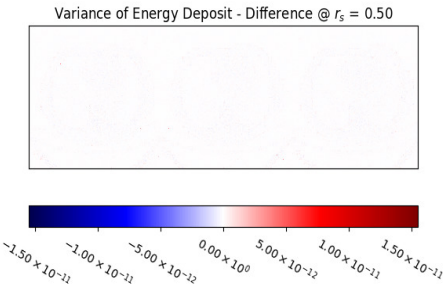


Figure B.56: Absolute Difference in Variance for Energy Deposit Tally between Figure B.54 and Figure B.53 (using min/max of data)

B.2.2.2 Number of Collisions

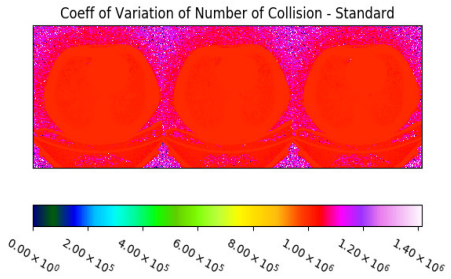


Figure B.57: Coefficient of Variation for Number of Collisions Tally — Standard

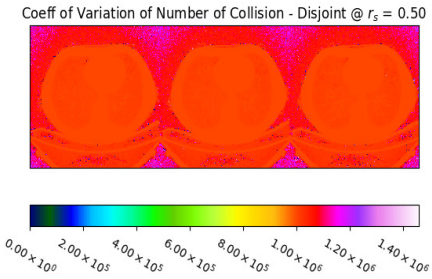


Figure B.58: Coefficient of Variation for Number of Collisions Tally — Disjoint

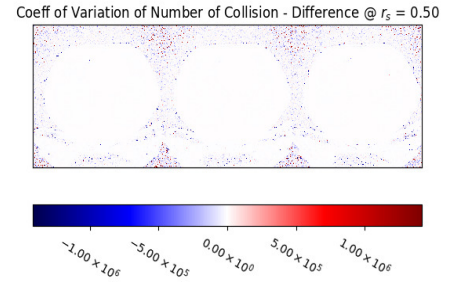


Figure B.59: Absolute Difference in Coefficient of Variation for Number of Collisions Tally between Figure B.58 and Figure B.57 (using min/max of tally)

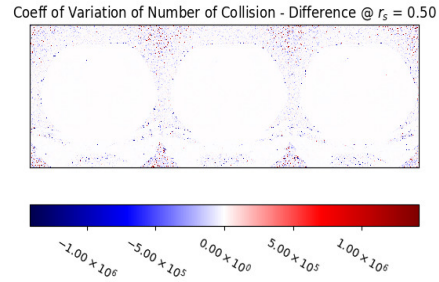


Figure B.60: Absolute Difference in Coefficient of Variation for Number of Collisions Tally between Figure B.58 and Figure B.57 (using min/max of data)

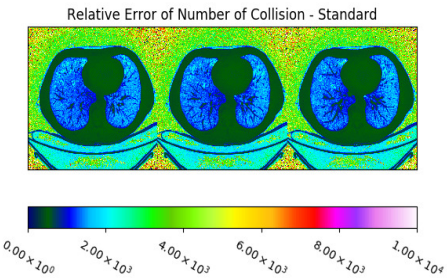


Figure B.61: Relative Error for Number of Collisions Tally — Standard

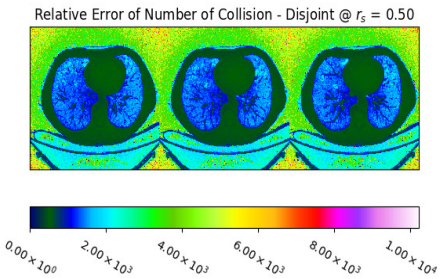


Figure B.62: Relative Error for Number of Collisions Tally — Disjoint

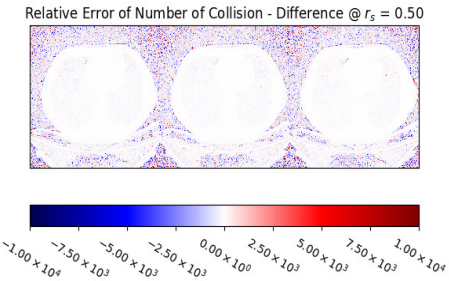


Figure B.63: Absolute Difference in Relative Error for Number of Collisions Tally between Figure B.62 and Figure B.61 (using min/max of tally)

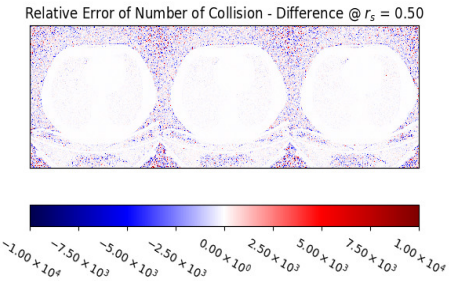


Figure B.64: Absolute Difference in Relative Error for Number of Collisions Tally between Figure B.62 and Figure B.61 (using min/max of data)

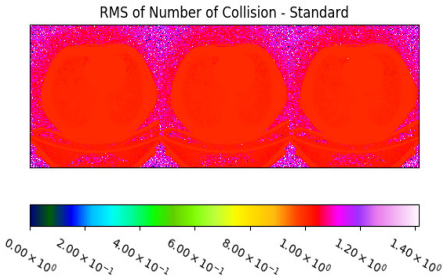


Figure B.65: Root Mean Squared (RMS) for Number of Collisions Tally — Standard

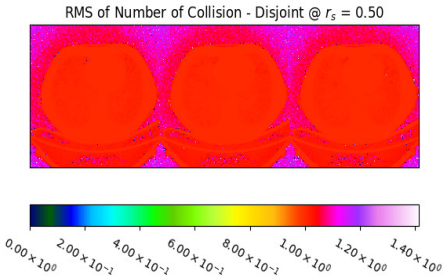


Figure B.66: Root Mean Squared (RMS) for Number of Collisions Tally — Disjoint

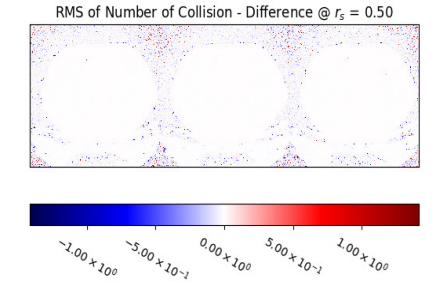


Figure B.67: Absolute Difference in Root Mean Squared (RMS) for Number of Collisions Tally between Figure B.66 and Figure B.65 (using min/max of tally)

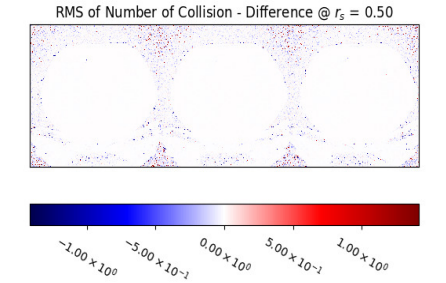


Figure B.68: Absolute Difference in Root Mean Squared (RMS) for Number of Collisions Tally between Figure B.66 and Figure B.65 (using min/max of data)

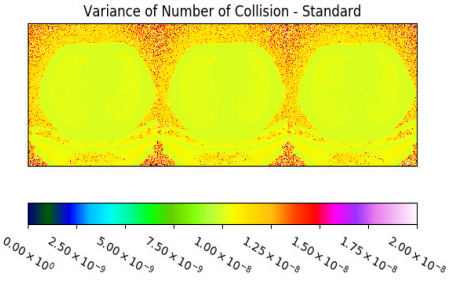


Figure B.69: Variance for Number of Collisions Tally — Standard

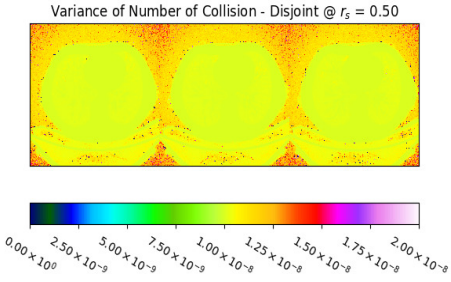


Figure B.70: Variance for Number of Collisions Tally — Disjoint

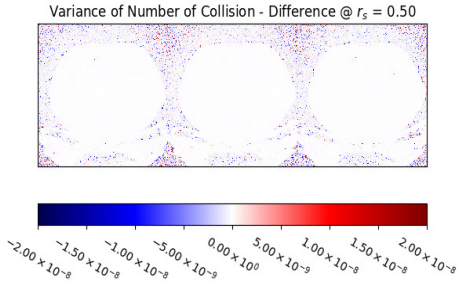


Figure B.71: Absolute Difference in Variance for Number of Collisions Tally between Figure B.70 and Figure B.69 (using min/max of tally)

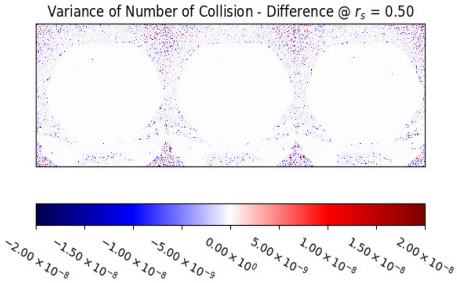


Figure B.72: Absolute Difference in Variance for Number of Collisions Tally between Figure B.70 and Figure B.69 (using min/max of data)

B.2.2.3 Termination

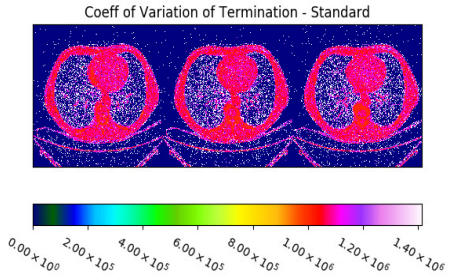


Figure B.73: Coefficient of Variation for Termination Tally — Standard

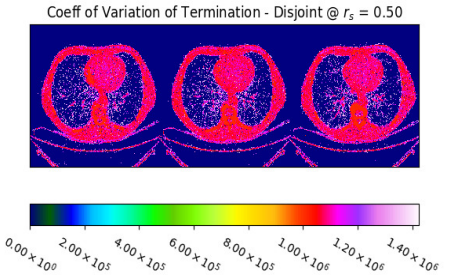


Figure B.74: Coefficient of Variation for Termination Tally — Disjoint

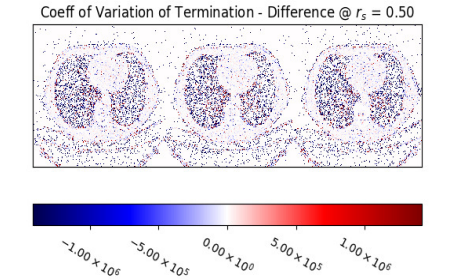


Figure B.75: Absolute Difference in Coefficient of Variation for Termination Tally between Figure B.74 and Figure B.73 (using min/max of tally)

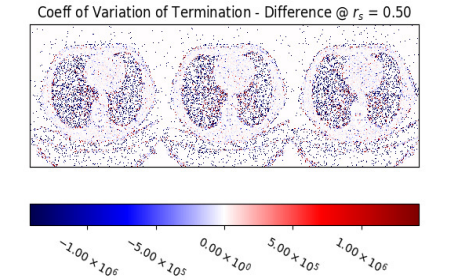


Figure B.76: Absolute Difference in Coefficient of Variation for Termination Tally between Figure B.74 and Figure B.73 (using min/max of data)

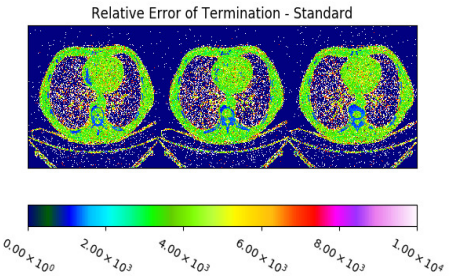


Figure B.77: Relative Error for Termination Tally — Standard

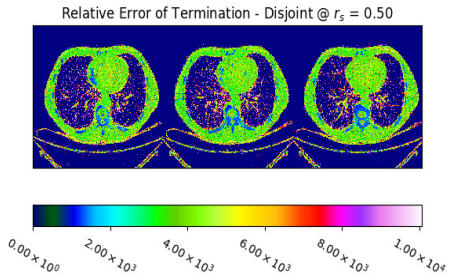


Figure B.78: Relative Error for Termination Tally — Disjoint

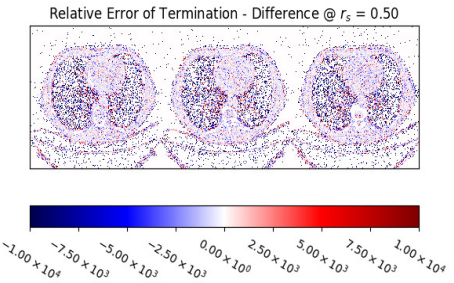


Figure B.79: Absolute Difference in Relative Error for Termination Tally between Figure B.78 and Figure B.77 (using min/max of tally)

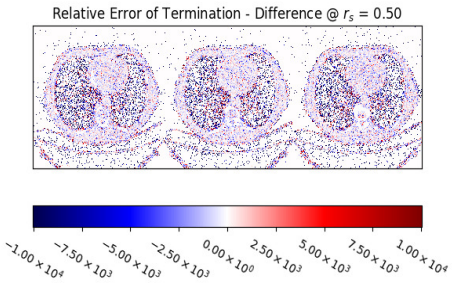


Figure B.80: Absolute Difference in Relative Error for Termination Tally between Figure B.78 and Figure B.77 (using min/max of data)

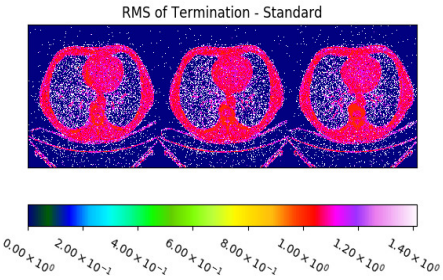


Figure B.81: Root Mean Squared (RMS) for Termination Tally — Standard

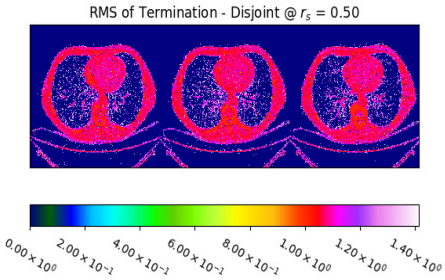


Figure B.82: Root Mean Squared (RMS) for Termination Tally — Disjoint

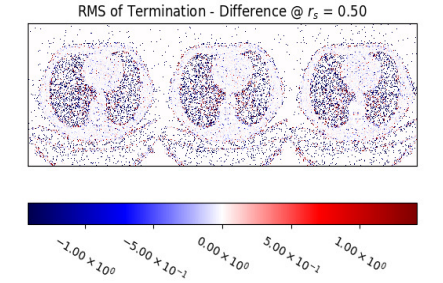


Figure B.83: Absolute Difference in Root Mean Squared (RMS) for Termination Tally between Figure B.82 and Figure B.81 (using min/max of tally)

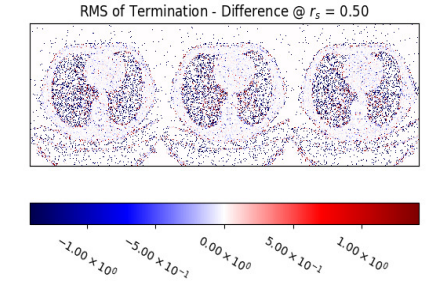


Figure B.84: Absolute Difference in Root Mean Squared (RMS) for Termination Tally between Figure B.82 and Figure B.81 (using min/max of data)

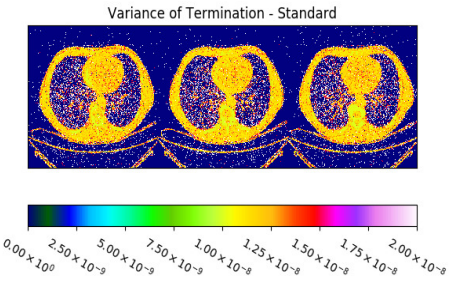


Figure B.85: Variance for Termination Tally — Standard

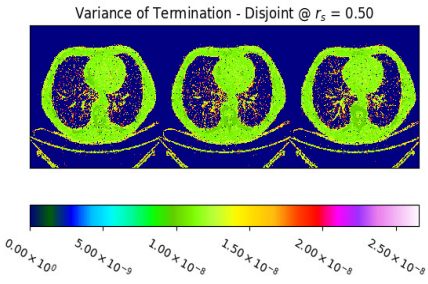


Figure B.86: Variance for Termination Tally — Disjoint

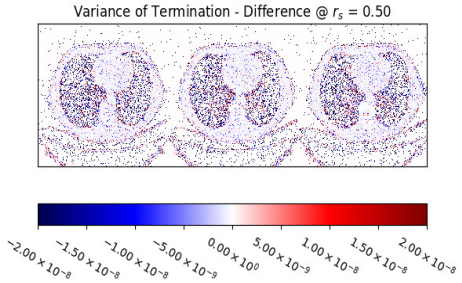


Figure B.87: Absolute Difference in Variance for Termination Tally between Figure B.86 and Figure B.85 (using min/max of tally)

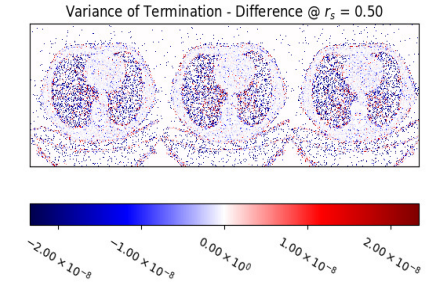


Figure B.88: Absolute Difference in Variance for Termination Tally between Figure B.86 and Figure B.85 (using min/max of data)

APPENDIX C

VARIOUS RECONSTRUCTION EXAMPLES

C.1 DICOM

The DICOM tally heatmaps in this appendix section were generated from linear distribution of 90-120 keV source photons after 1×10^7 particles. Geometry rendered from two contiguous DICOM CT slices (slice #1 and slice #2 from Section 3.16). The disjoint and standard tallies scored onto a $512 \times 512 \times 2$ mesh and the disjoint tallies were reconstructed in $16 \times 16 \times 2$ sub-blocks. The subrates used in this section was were 0.3, 0.5, and 0.7.

C.1.1 Cell Charge

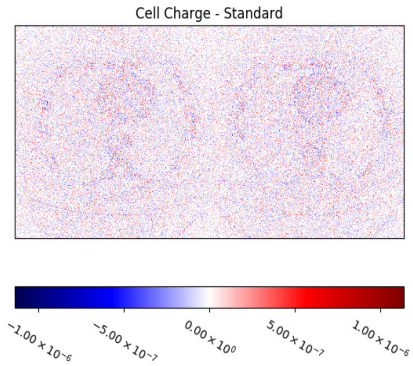


Figure C.1: Cell Charge Tally using standard tallies

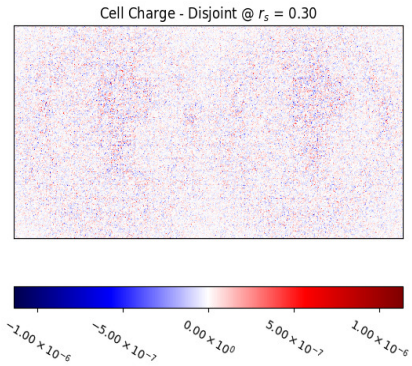


Figure C.2: Cell Charge Tally using disjoint tallies. $r_s = 0.3$

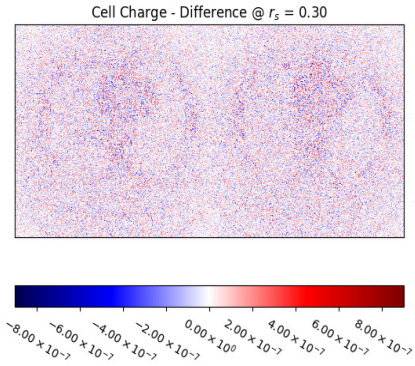


Figure C.3: Absolute Difference in Cell Charge Tally between Figure C.1 and Figure C.2 (using min/max of tally)

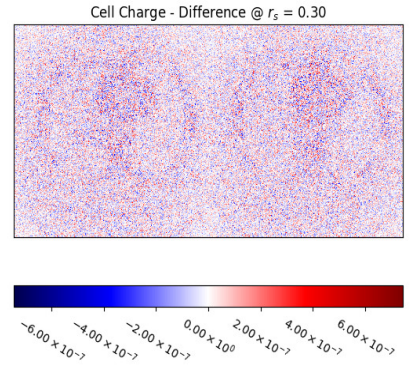


Figure C.4: Absolute Difference in Cell Charge Tally between Figure C.1 and Figure C.2 (using min/max of data)

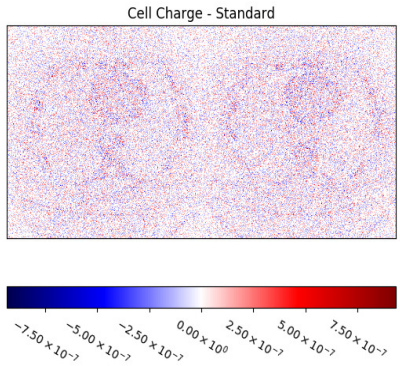


Figure C.5: Cell Charge Tally using standard tallies

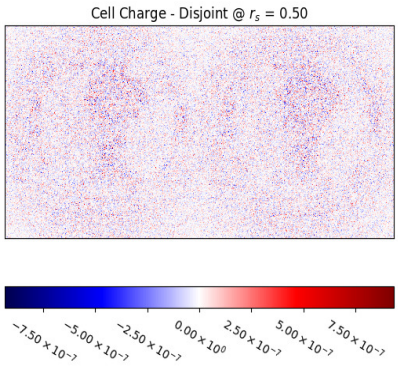


Figure C.6: Cell Charge Tally using disjoint tallies. $r_s = 0.5$

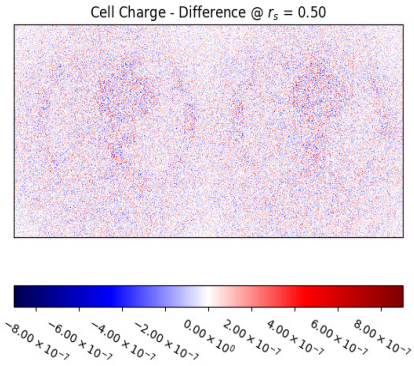


Figure C.7: Absolute Difference in Cell Charge Tally between Figure C.5 and Figure C.6 (using min/max of tally)

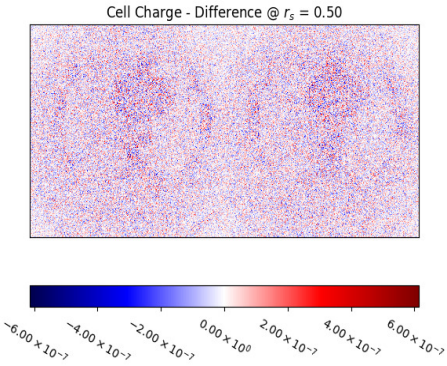


Figure C.8: Absolute Difference in Cell Charge Tally between Figure C.5 and Figure C.6 (using min/max of data)

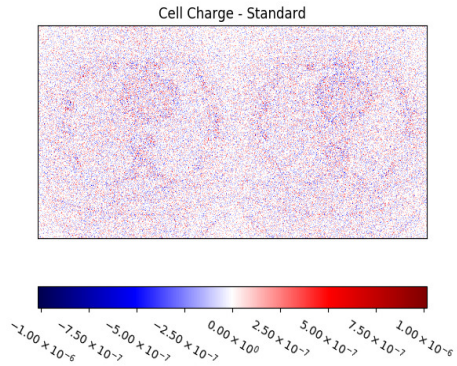


Figure C.9: Cell Charge Tally using standard tallies

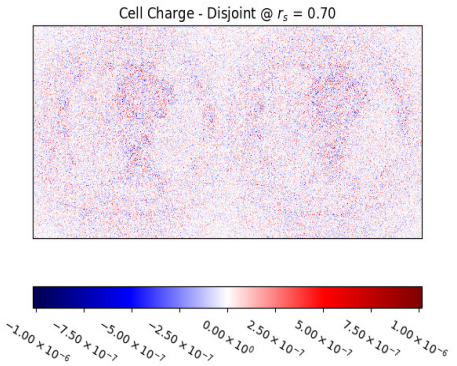


Figure C.10: Cell Charge Tally using disjoint tallies. $r_s = 0.7$

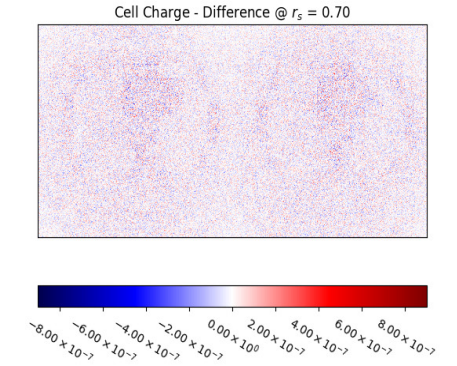


Figure C.11: Absolute Difference in Cell Charge Tally between Figure C.9 and Figure C.10 (using min/max of tally)

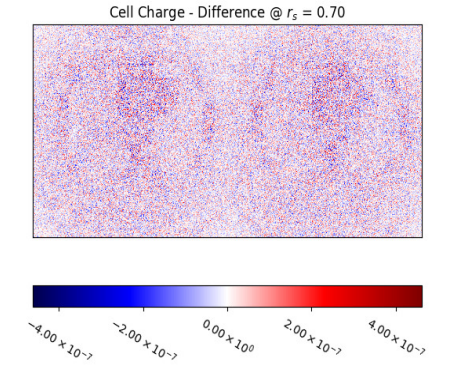


Figure C.12: Absolute Difference in Cell Charge Tally between Figure C.9 and Figure C.10 (using min/max of data)

C.1.2 Dose Deposit

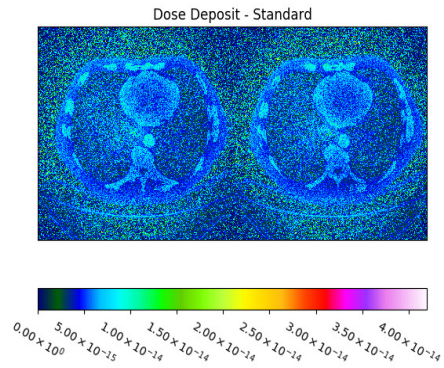


Figure C.13: Dose Deposit Tally using standard tallies

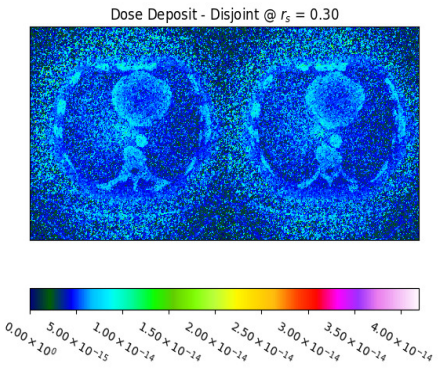


Figure C.14: Dose Deposit Tally using disjoint tallies. $r_s = 0.3$

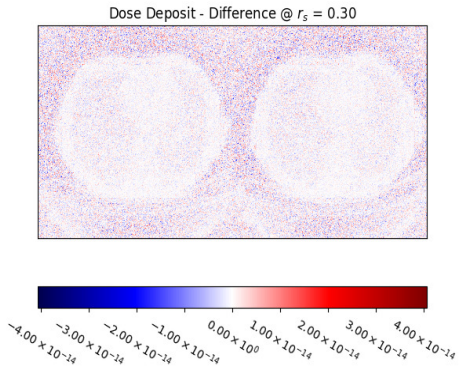


Figure C.15: Absolute Difference in Dose Deposit Tally between Figure C.13 and Figure C.14 (using min/max of tally)

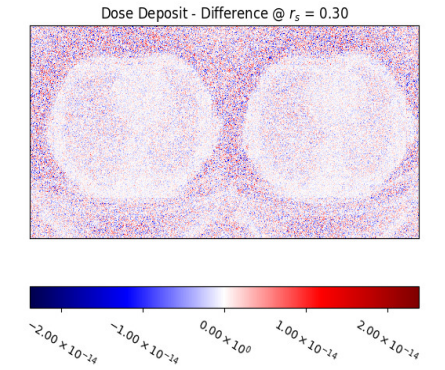


Figure C.16: Absolute Difference in Dose Deposit Tally between Figure C.13 and Figure C.14 (using min/max of data)

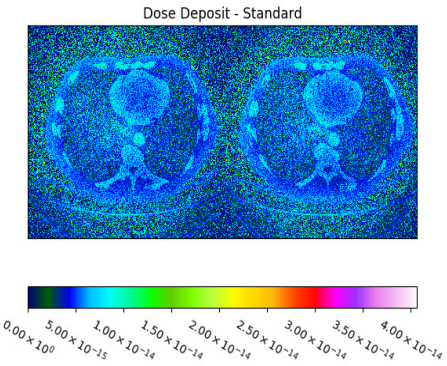


Figure C.17: Dose Deposit Tally using standard tallies

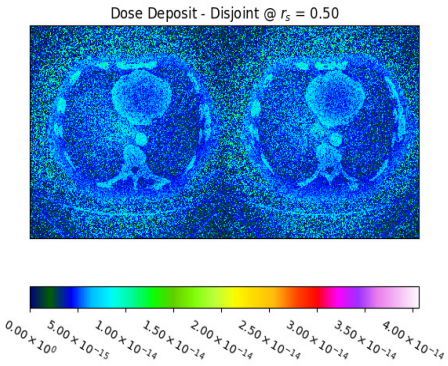


Figure C.18: Dose Deposit Tally using disjoint tallies. $r_s = 0.5$

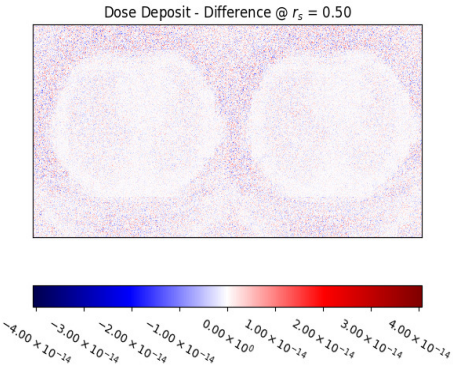


Figure C.19: Absolute Difference in Dose Deposit Tally between Figure C.17 and Figure C.18 (using min/max of tally)

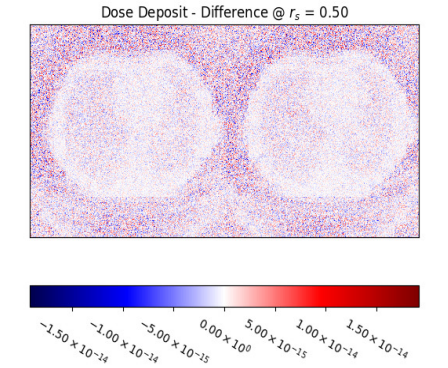


Figure C.20: Absolute Difference in Dose Deposit Tally between Figure C.17 and Figure C.18 (using min/max of data)

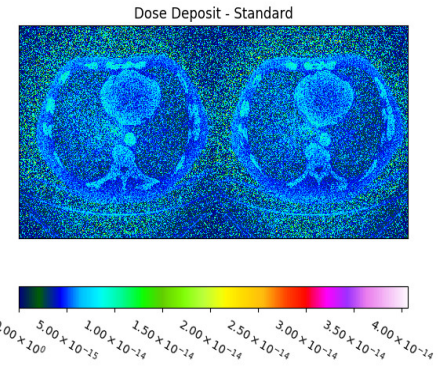


Figure C.21: Dose Deposit Tally using standard tallies

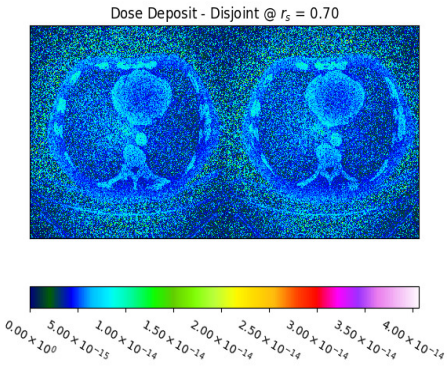


Figure C.22: Dose Deposit Tally using disjoint tallies. $r_s = 0.7$

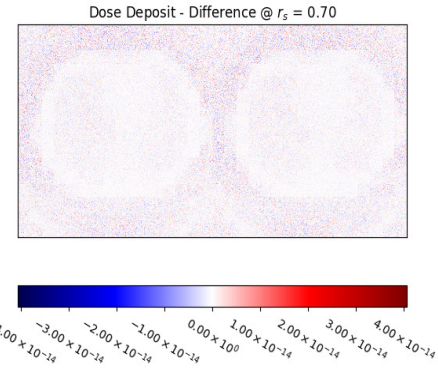


Figure C.23: Absolute Difference in Dose Deposit Tally between Figure C.21 and Figure C.22 (using min/max of tally)

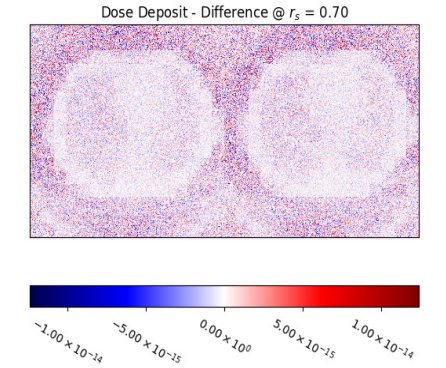


Figure C.24: Absolute Difference in Dose Deposit Tally between Figure C.21 and Figure C.22 (using min/max of data)

C.1.3 Energy Deposit

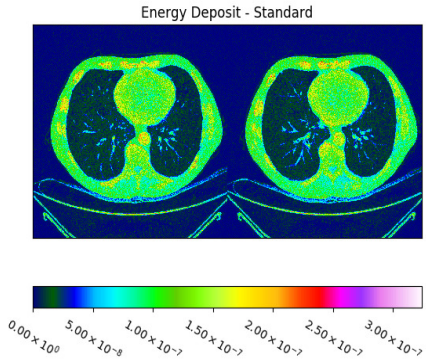


Figure C.25: Energy Deposit Tally using standard tallies

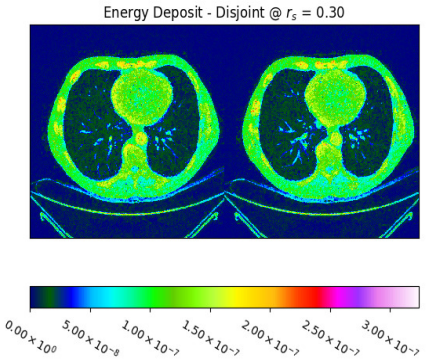


Figure C.26: Energy Deposit Tally using disjoint tallies. $r_s = 0.3$

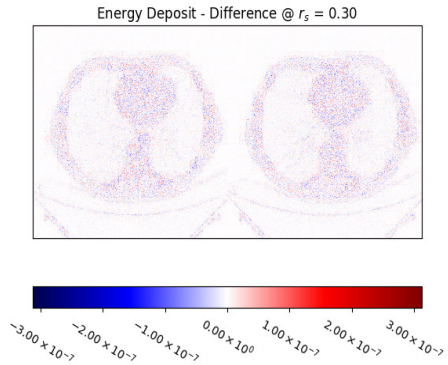


Figure C.27: Absolute Difference in Energy Deposit Tally between Figure C.25 and Figure C.26 (using min/max of tally)

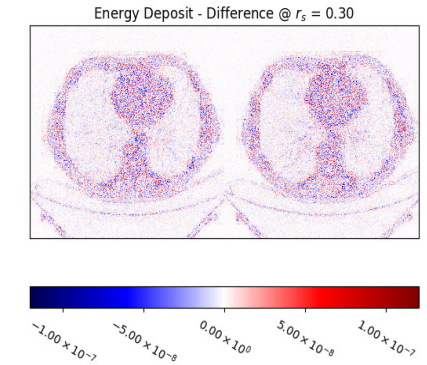


Figure C.28: Absolute Difference in Energy Deposit Tally between Figure C.25 and Figure C.26 (using min/max of data)

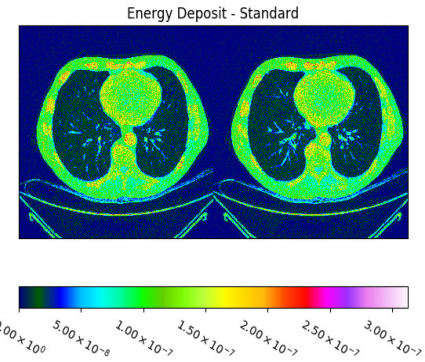


Figure C.29: Energy Deposit Tally using standard tallies

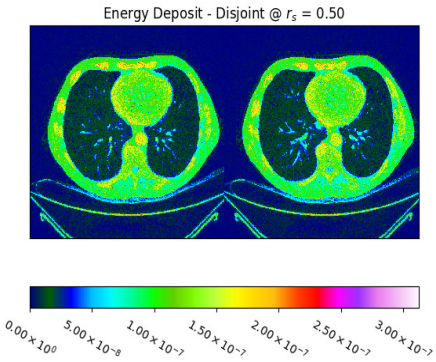


Figure C.30: Energy Deposit Tally using disjoint tallies. $r_s = 0.5$

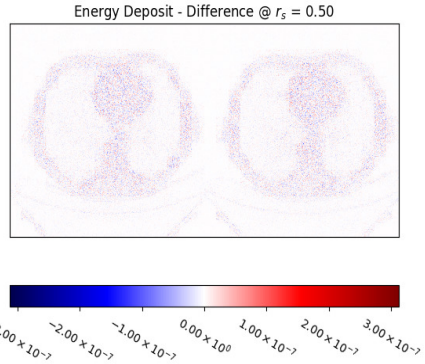


Figure C.31: Absolute Difference in Energy Deposit Tally between Figure C.29 and Figure C.30 (using min/max of tally)

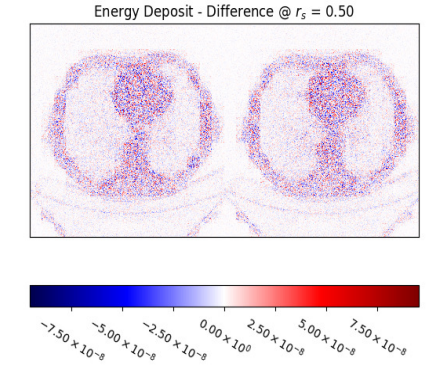


Figure C.32: Absolute Difference in Energy Deposit Tally between Figure C.29 and Figure C.30 (using min/max of data)

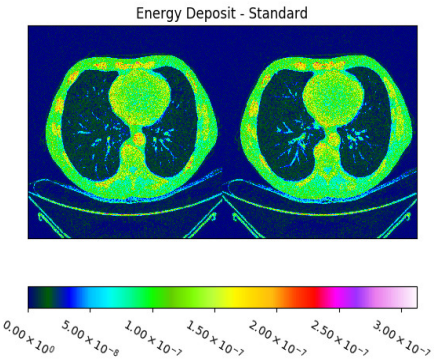


Figure C.33: Energy Deposit Tally using standard tallies

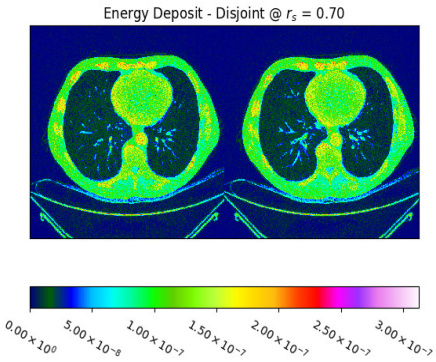


Figure C.34: Energy Deposit Tally using disjoint tallies. $r_s = 0.7$

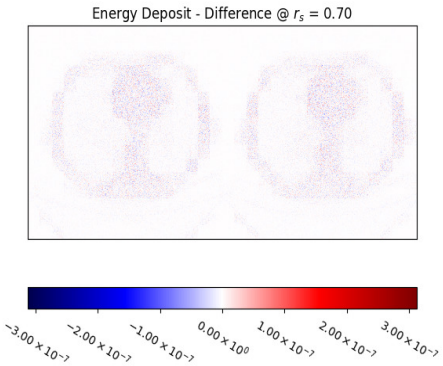


Figure C.35: Absolute Difference in Energy Deposit Tally between Figure C.33 and Figure C.34 (using min/max of tally)

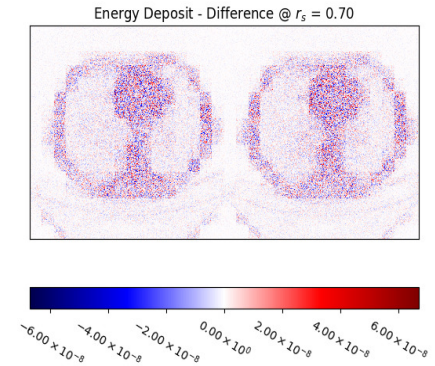


Figure C.36: Absolute Difference in Energy Deposit Tally between Figure C.33 and Figure C.34 (using min/max of data)

C.1.4 Flat Surface Current

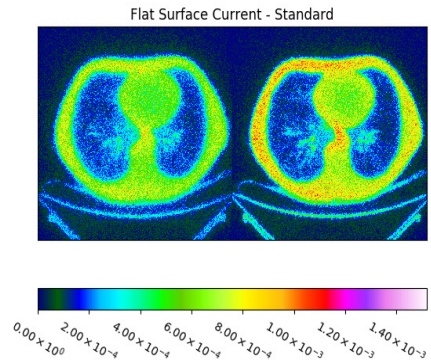


Figure C.37: Flat Surface Current Tally using standard tallies

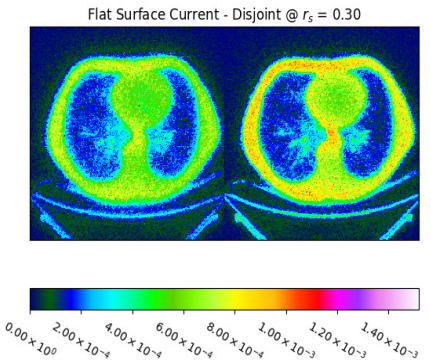


Figure C.38: Flat Surface Current Tally using disjoint tallies. $r_s = 0.3$

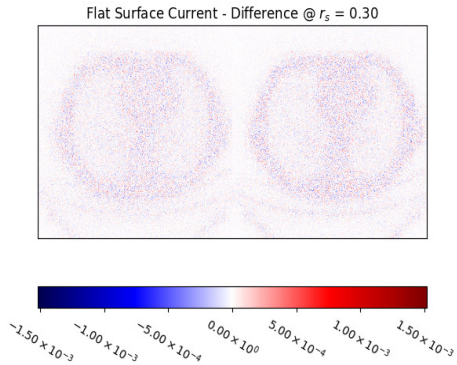


Figure C.39: Absolute Difference in Flat Surface Current Tally between Figure C.37 and Figure C.38 (using min/max of tally)

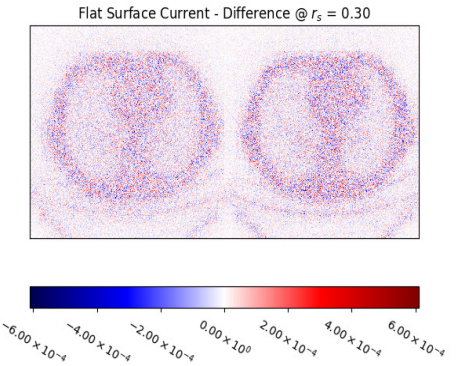


Figure C.40: Absolute Difference in Flat Surface Current Tally between Figure C.37 and Figure C.38 (using min/max of data)

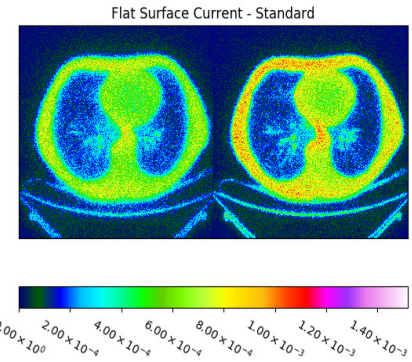


Figure C.41: Flat Surface Current Tally using standard tallies

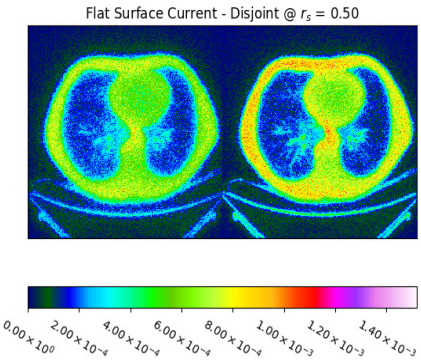


Figure C.42: Flat Surface Current Tally using disjoint tallies. $r_s = 0.5$

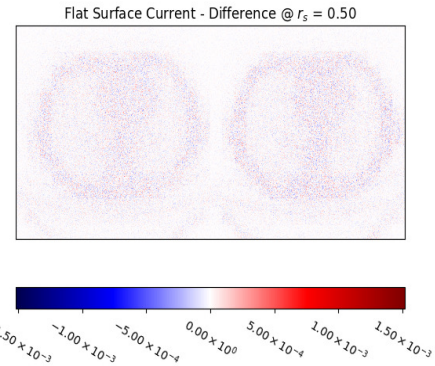


Figure C.43: Absolute Difference in Flat Surface Current Tally between Figure C.41 and Figure C.42 (using min/max of tally)

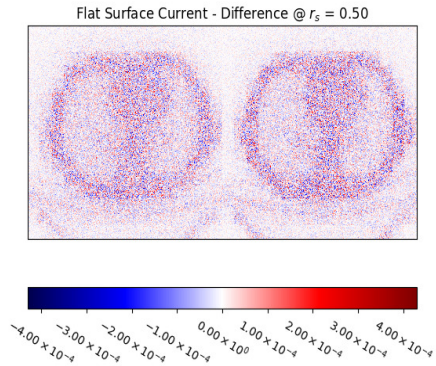


Figure C.44: Absolute Difference in Flat Surface Current Tally between Figure C.41 and Figure C.42 (using min/max of data)

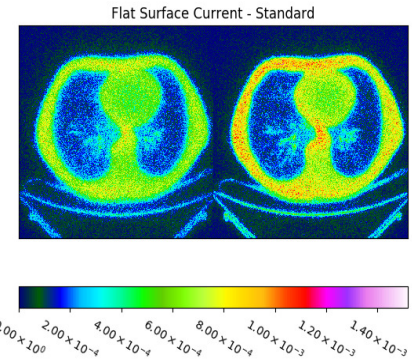


Figure C.45: Flat Surface Current Tally using standard tallies

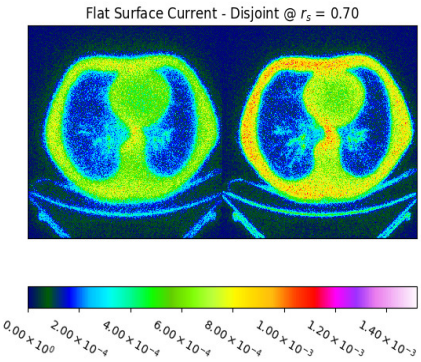


Figure C.46: Flat Surface Current Tally using disjoint tallies. $r_s = 0.7$

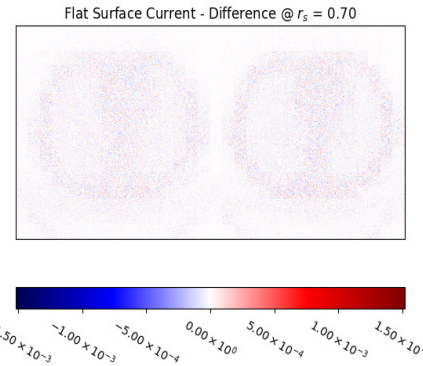


Figure C.47: Absolute Difference in Flat Surface Current Tally between Figure C.45 and Figure C.46 (using min/max of tally)

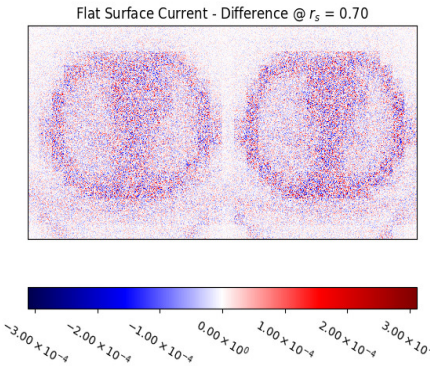


Figure C.48: Absolute Difference in Flat Surface Current Tally between Figure C.45 and Figure C.46 (using min/max of data)

C.1.5 Number of Collision

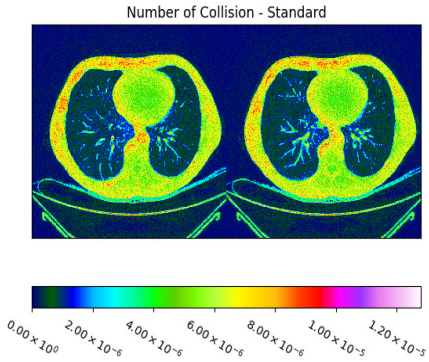


Figure C.49: Number of Collision Tally using standard tallies

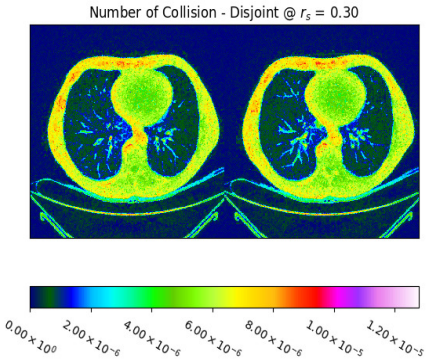


Figure C.50: Number of Collision Tally using disjoint tallies. $r_s = 0.3$

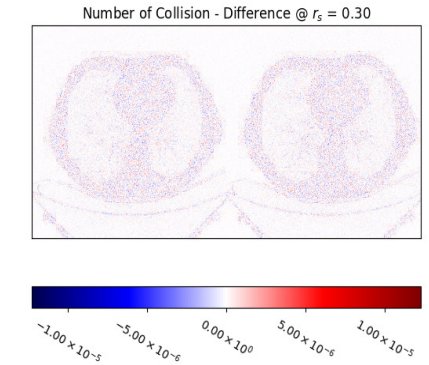


Figure C.51: Absolute Difference in Number of Collision Tally between Figure C.49 and Figure C.50 (using min/max of tally)

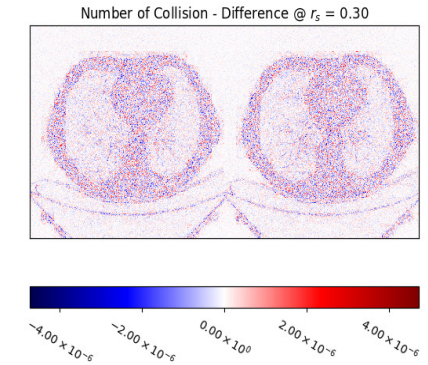


Figure C.52: Absolute Difference in Number of Collision Tally between Figure C.49 and Figure C.50 (using min/max of data)

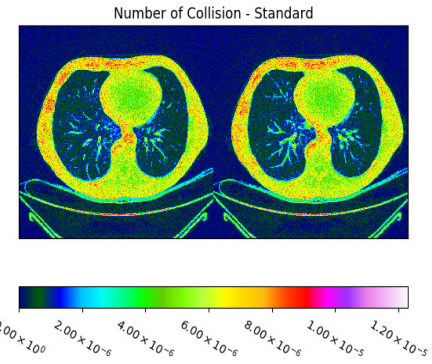


Figure C.53: Number of Collision Tally using standard tallies

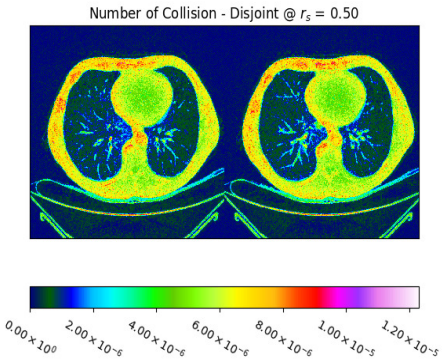


Figure C.54: Number of Collision Tally using disjoint tallies. $r_s = 0.5$

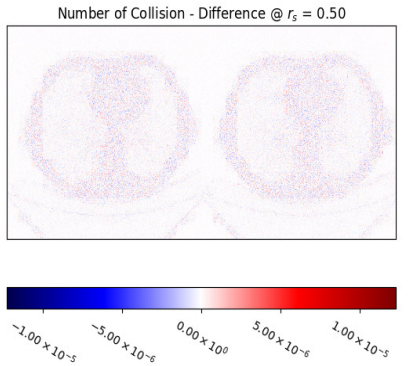


Figure C.55: Absolute Difference in Number of Collision Tally between Figure C.53 and Figure C.54 (using min/max of tally)

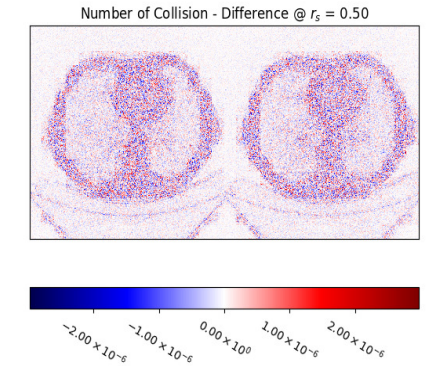


Figure C.56: Absolute Difference in Number of Collision Tally between Figure C.53 and Figure C.54 (using min/max of data)

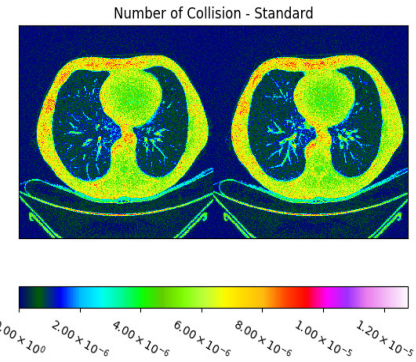


Figure C.57: Number of Collision Tally using standard tallies

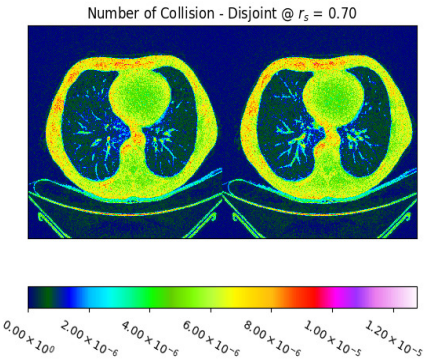


Figure C.58: Number of Collision Tally using disjoint tallies. $r_s = 0.7$

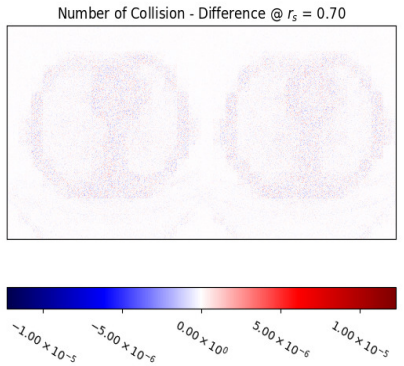


Figure C.59: Absolute Difference in Number of Collision Tally between Figure C.57 and Figure C.58 (using min/max of tally)

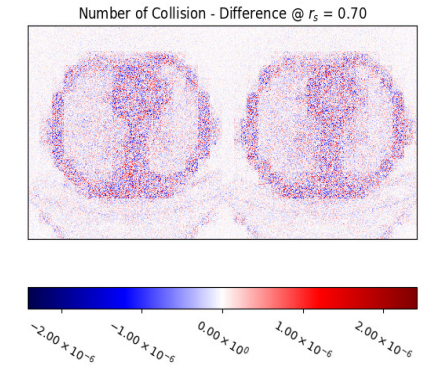


Figure C.60: Absolute Difference in Number of Collision Tally between Figure C.57 and Figure C.58 (using min/max of data)

C.1.6 Number of Secondary

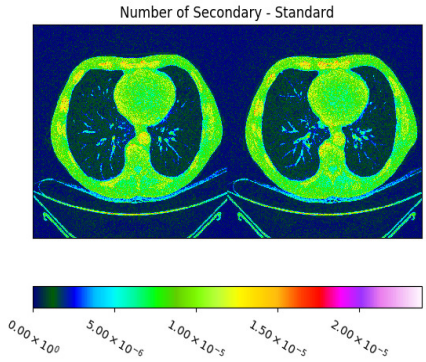


Figure C.61: Number of Secondary Tally using standard tallies

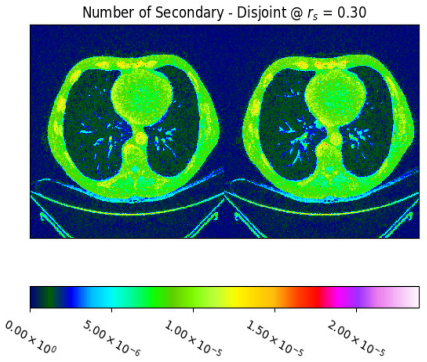


Figure C.62: Number of Secondary Tally using disjoint tallies. $r_s = 0.3$

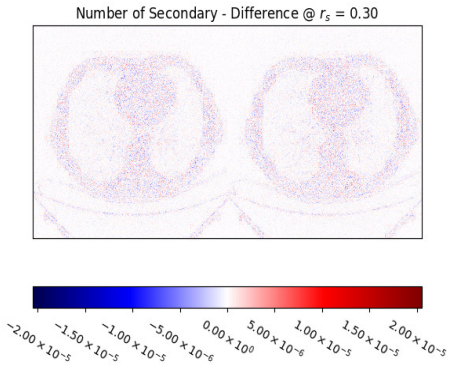


Figure C.63: Absolute Difference in Number of Secondary Tally between Figure C.61 and Figure C.62 (using min/max of tally)

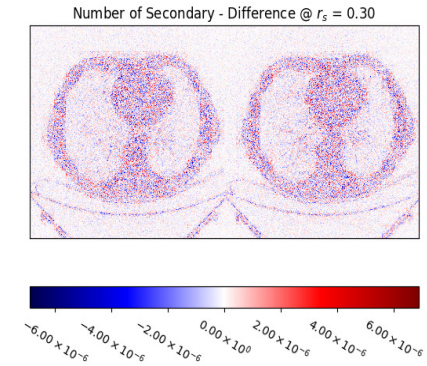


Figure C.64: Absolute Difference in Number of Secondary Tally between Figure C.61 and Figure C.62 (using min/max of data)

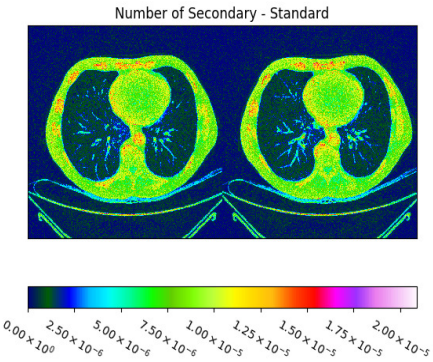


Figure C.65: Number of Secondary Tally using standard tallies

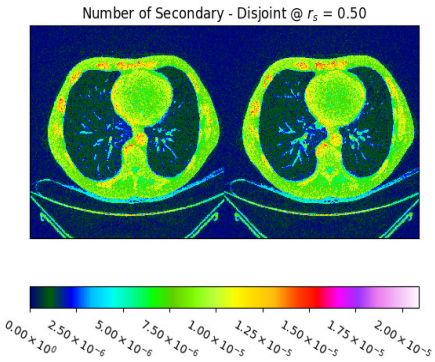


Figure C.66: Number of Secondary Tally using disjoint tallies. $r_s = 0.5$

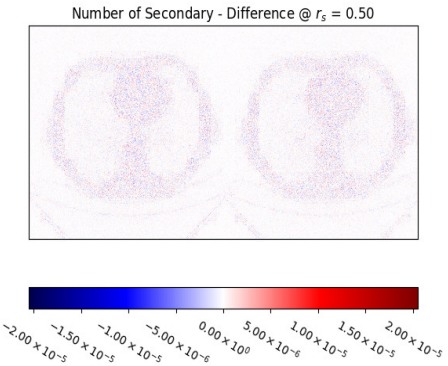


Figure C.67: Absolute Difference in Number of Secondary Tally between Figure C.65 and Figure C.66 (using min/max of tally)

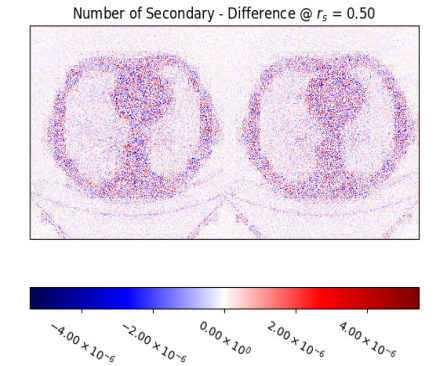


Figure C.68: Absolute Difference in Number of Secondary Tally between Figure C.65 and Figure C.66 (using min/max of data)

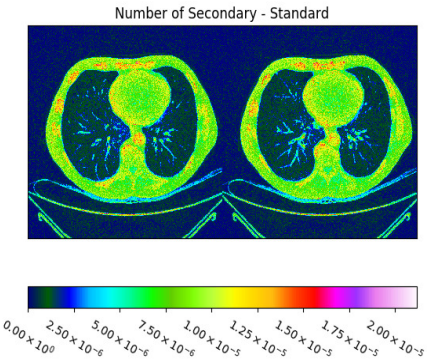


Figure C.69: Number of Secondary Tally using standard tallies

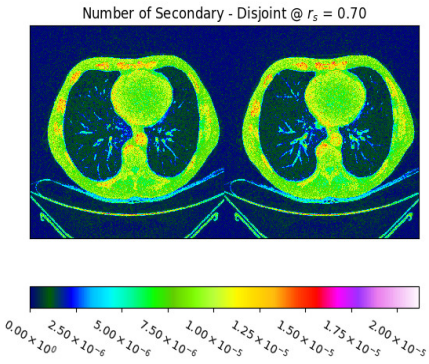


Figure C.70: Number of Secondary Tally using disjoint tallies. $r_s = 0.7$

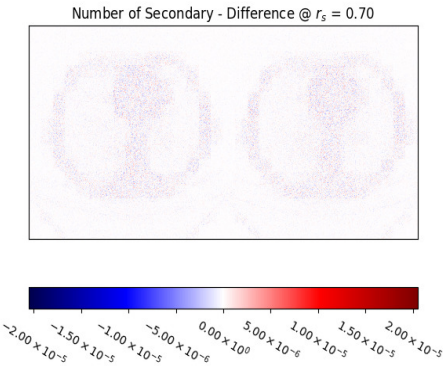


Figure C.71: Absolute Difference in Number of Secondary Tally between Figure C.69 and Figure C.70 (using min/max of tally)

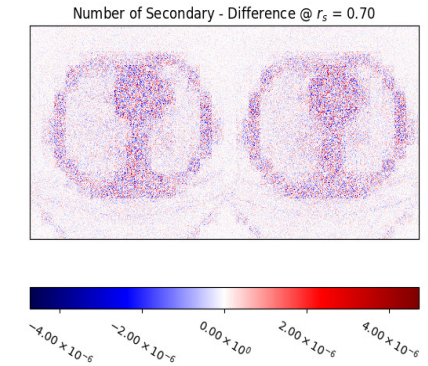


Figure C.72: Absolute Difference in Number of Secondary Tally between Figure C.69 and Figure C.70 (using min/max of data)

C.1.7 Population

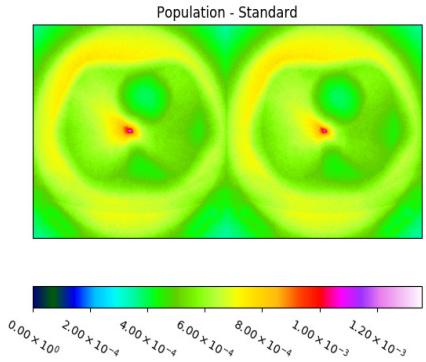


Figure C.73: Population Tally using standard tallies

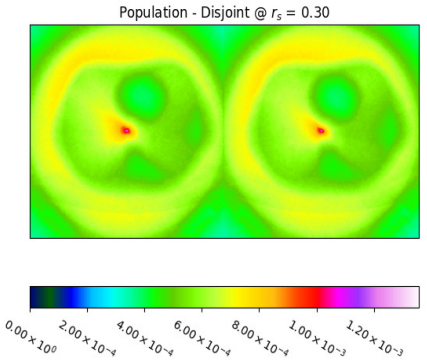


Figure C.74: Population Tally using disjoint tallies. $r_s = 0.3$

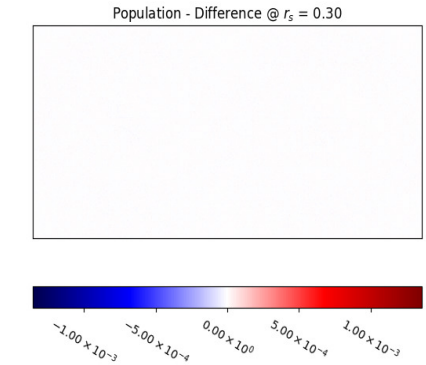


Figure C.75: Absolute Difference in Population Tally between Figure C.73 and Figure C.74 (using min/max of tally)

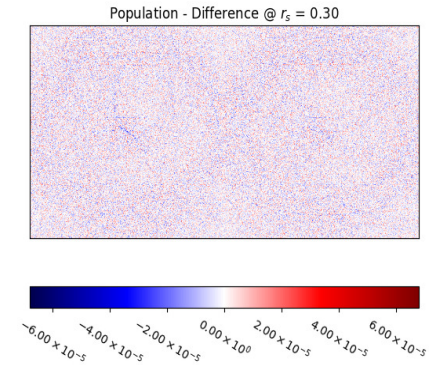


Figure C.76: Absolute Difference in Population Tally between Figure C.73 and Figure C.74 (using min/max of data)

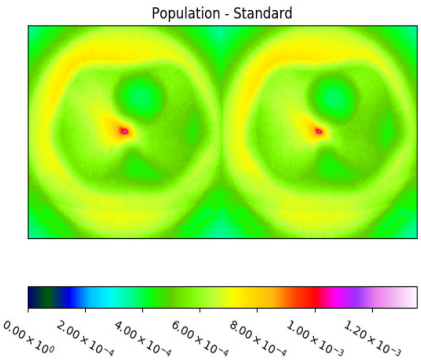


Figure C.77: Population Tally using standard tallies

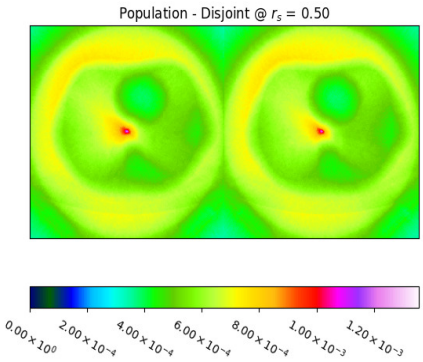


Figure C.78: Population Tally using disjoint tallies. $r_s = 0.5$

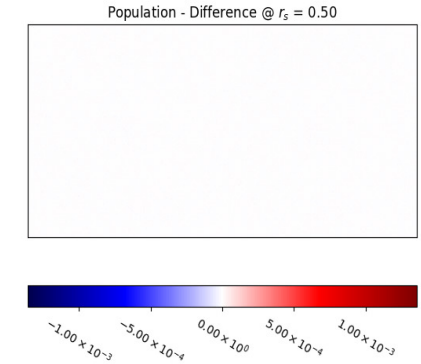


Figure C.79: Absolute Difference in Population Tally between Figure C.77 and Figure C.78 (using min/max of tally)

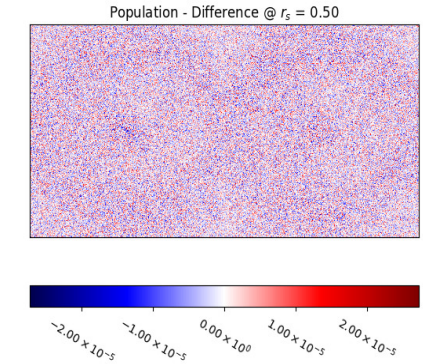


Figure C.80: Absolute Difference in Population Tally between Figure C.77 and Figure C.78 (using min/max of data)

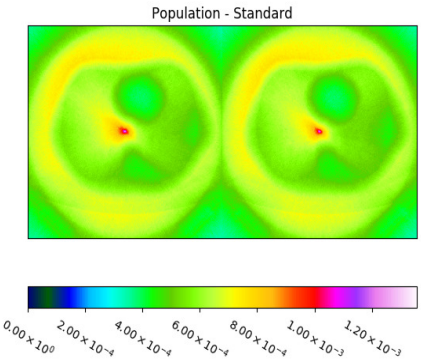


Figure C.81: Population Tally using standard tallies

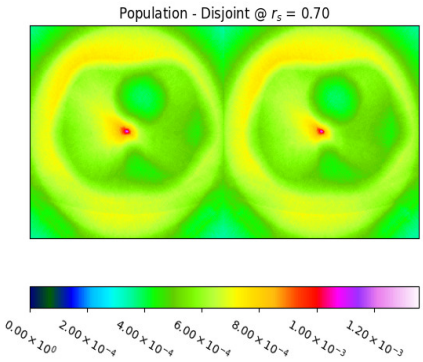


Figure C.82: Population Tally using disjoint tallies. $r_s = 0.7$

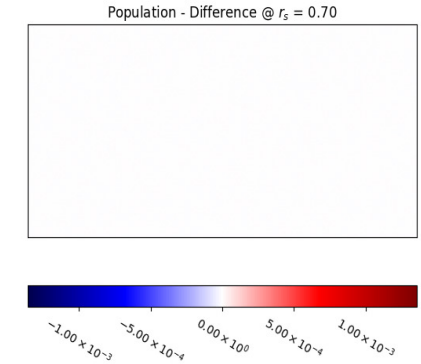


Figure C.83: Absolute Difference in Population Tally between Figure C.81 and Figure C.82 (using min/max of tally)

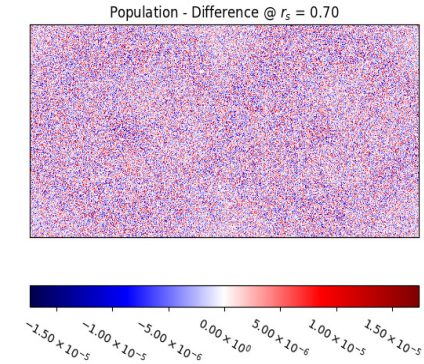


Figure C.84: Absolute Difference in Population Tally between Figure C.81 and Figure C.82 (using min/max of data)

C.1.8 Termination

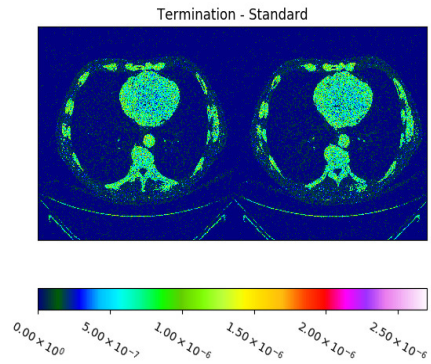


Figure C.85: Termination Tally using standard tallies

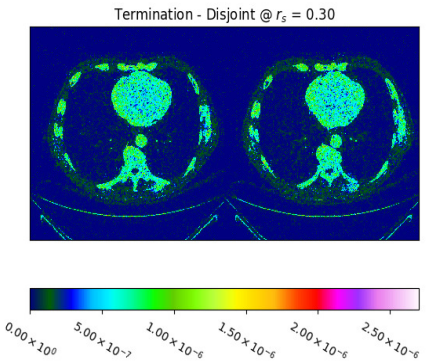


Figure C.86: Termination Tally using disjoint tallies. $r_s = 0.3$

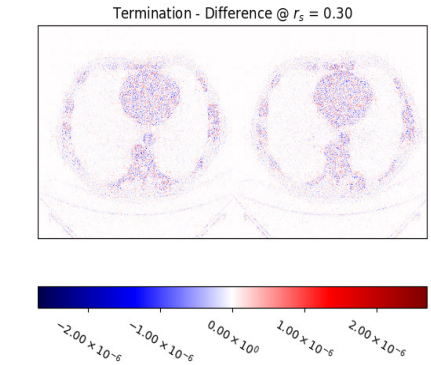


Figure C.87: Absolute Difference in Termination Tally between Figure C.85 and Figure C.86 (using min/max of tally)

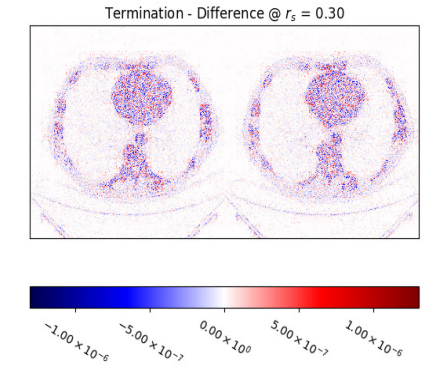


Figure C.88: Absolute Difference in Termination Tally between Figure C.85 and Figure C.86 (using min/max of data)

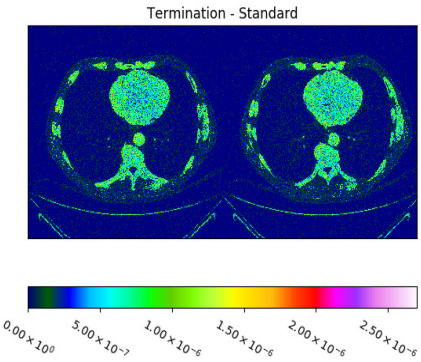


Figure C.89: Termination Tally using standard tallies

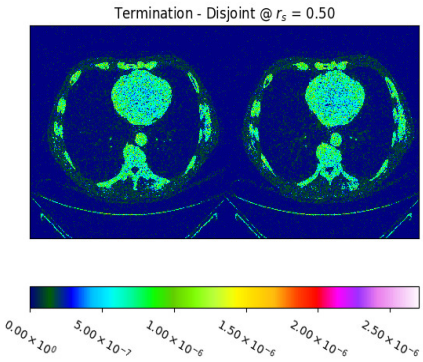


Figure C.90: Termination Tally using disjoint tallies. $r_s = 0.5$

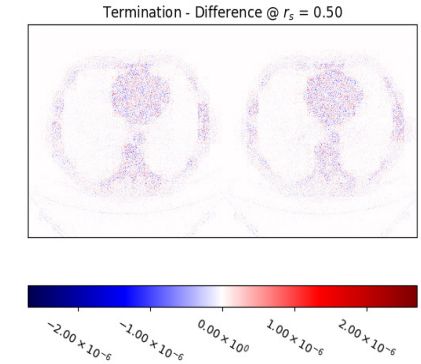


Figure C.91: Absolute Difference in Termination Tally between Figure C.89 and Figure C.90 (using min/max of tally)

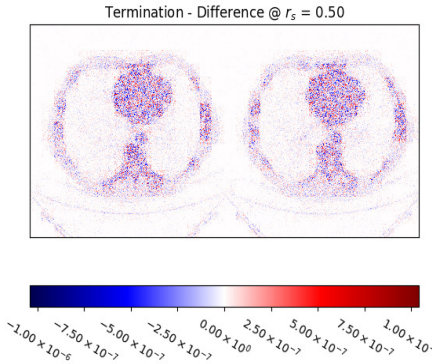


Figure C.92: Absolute Difference in Termination Tally between Figure C.89 and Figure C.90 (using min/max of data)

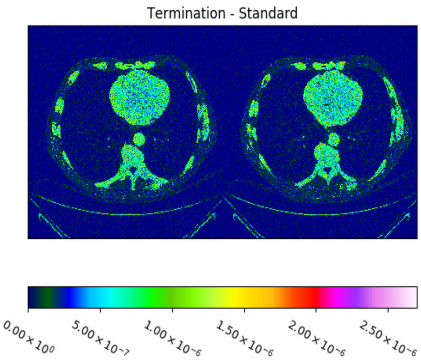


Figure C.93: Termination Tally using standard tallies

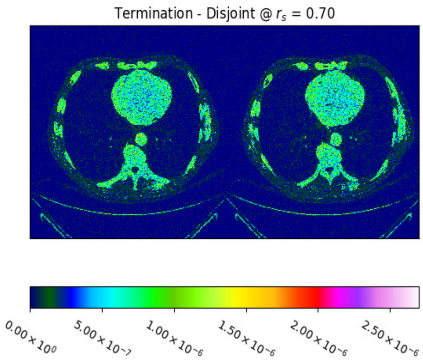


Figure C.94: Termination Tally using disjoint tallies. $r_s = 0.7$

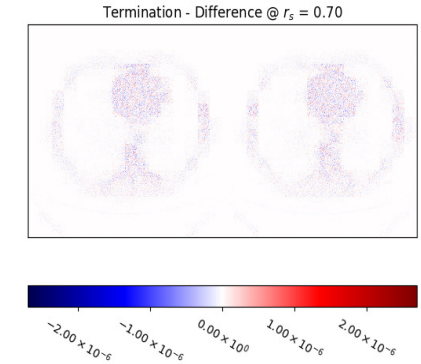


Figure C.95: Absolute Difference in Termination Tally between Figure C.93 and Figure C.94 (using min/max of tally)

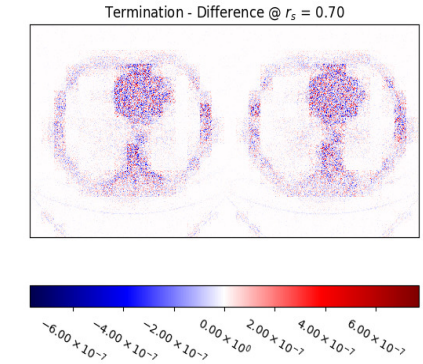


Figure C.96: Absolute Difference in Termination Tally between Figure C.93 and Figure C.94 (using min/max of data)

C.2 Reactor

The tally heatmaps in this appendix section were generated from random sampling of e^{-}, n^{0}, and γ^{0} after 1×10^{5} particles. The reactor-type geometry specifications are detailed in Section 3.4.1. The disjoint and standard tallies scored onto a $512 \times 512 \times 2$ mesh and the disjoint tallies were reconstructed in $16 \times 16 \times 2$ sub-blocks. The subrates used in this section was 0.25, 0.50, and 0.75.

C.2.1 Cell Charge

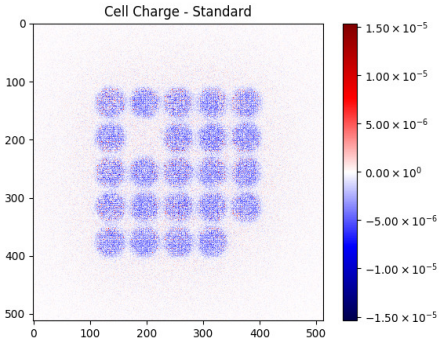


Figure C.97: Cell Charge Tally using standard tallies

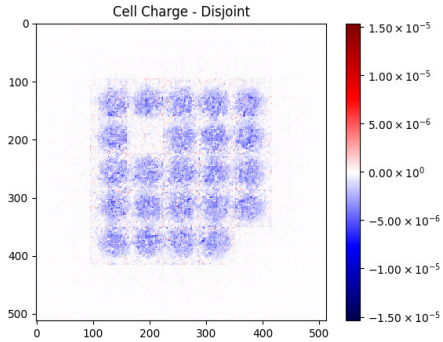


Figure C.98: Cell Charge Tally using disjoint tallies. $r_s = 0.25$

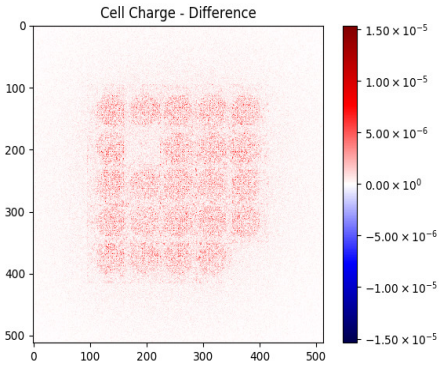


Figure C.99: Absolute Difference in Cell Charge Tally between Figure C.97 and Figure C.98 (using min/max of tally)

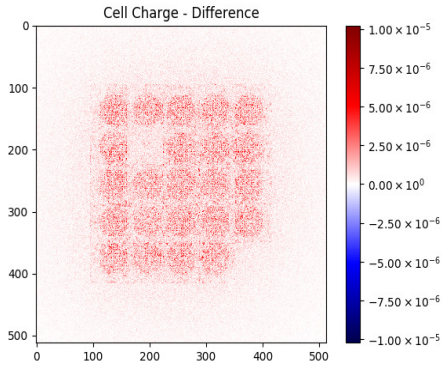


Figure C.100: Absolute Difference in Cell Charge Tally between Figure C.97 and Figure C.98 (using min/max of data)

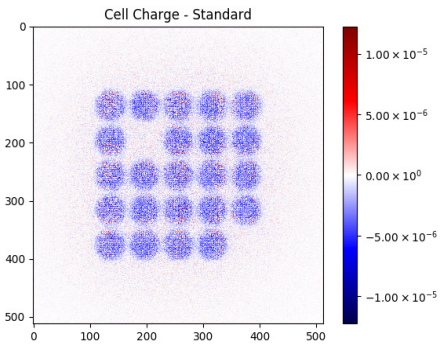


Figure C.101: Cell Charge Tally using standard tallies

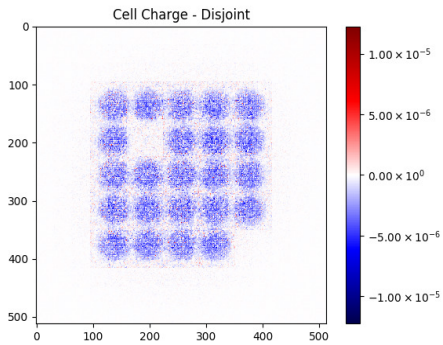


Figure C.102: Cell Charge Tally using disjoint tallies. $r_s = 0.50$

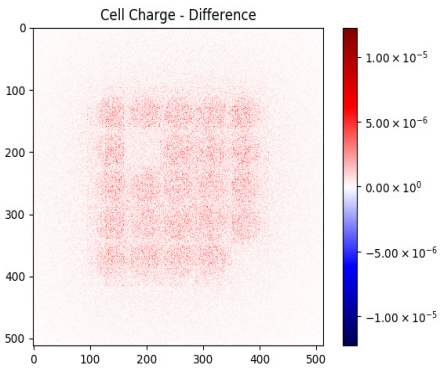


Figure C.103: Absolute Difference in Cell Charge Tally between Figure C.101 and Figure C.102 (using min/max of tally)

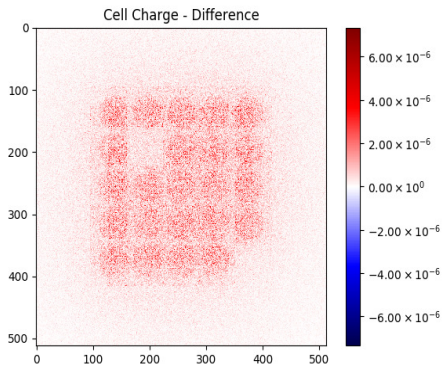


Figure C.104: Absolute Difference in Cell Charge Tally between Figure C.101 and Figure C.102 (using min/max of data)

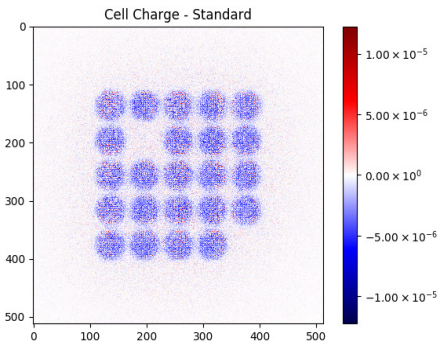


Figure C.105: Cell Charge Tally using standard tallies

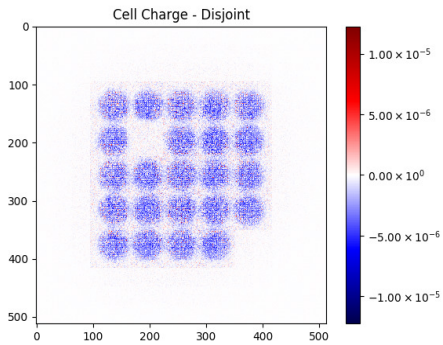


Figure C.106: Cell Charge Tally using disjoint tallies. $r_s = 0.75$

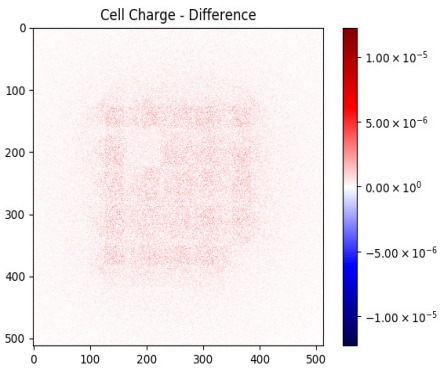


Figure C.107: Absolute Difference in Cell Charge Tally between Figure C.105 and Figure C.106 (using min/max of tally)

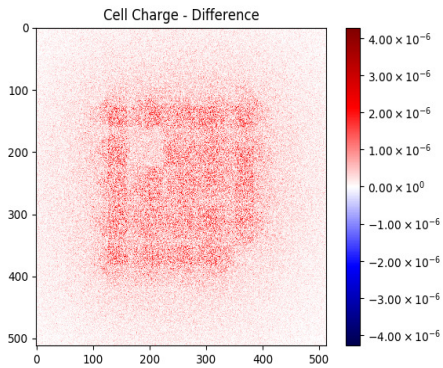


Figure C.108: Absolute Difference in Cell Charge Tally between Figure C.105 and Figure C.106 (using min/max of data)

C.2.2 Cell Flux

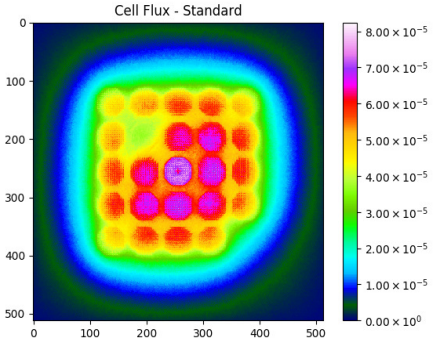


Figure C.109: Cell Flux Tally using standard tallies

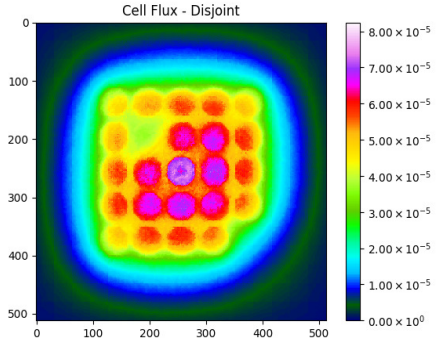


Figure C.110: Cell Flux Tally using disjoint tallies. $r_s = 0.25$

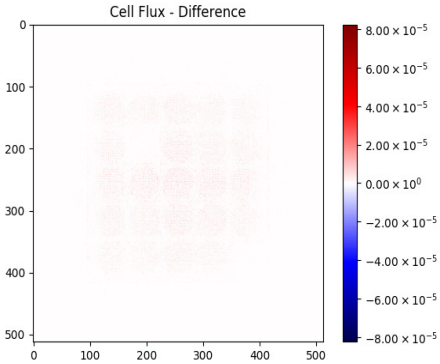


Figure C.111: Absolute Difference in Cell Flux Tally between Figure C.109 and Figure C.110 (using min/max of tally)

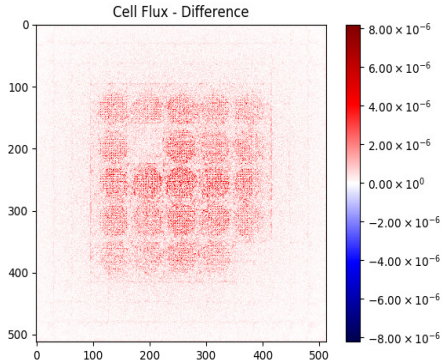


Figure C.112: Absolute Difference in Cell Flux Tally between Figure C.109 and Figure C.110 (using min/max of data)

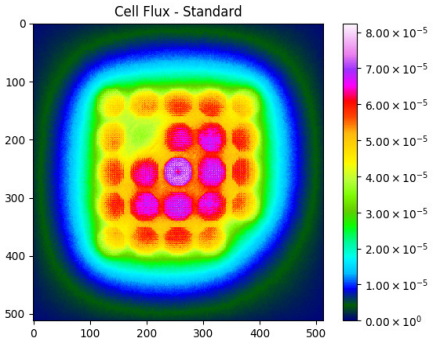


Figure C.113: Cell Flux Tally using standard tallies

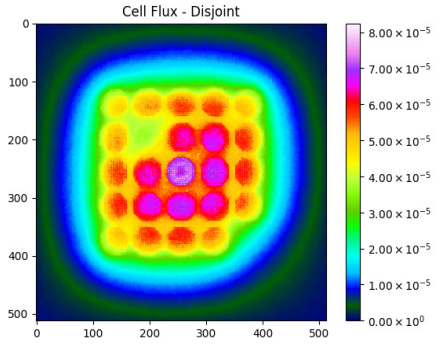


Figure C.114: Cell Flux Tally using disjoint tallies. $r_s = 0.50$

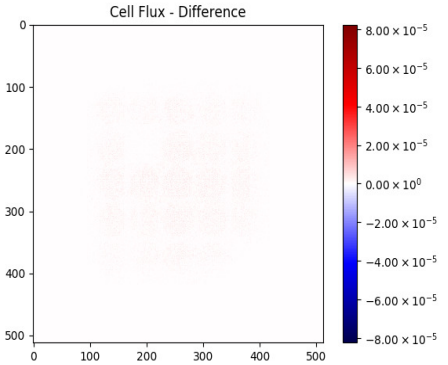


Figure C.115: Absolute Difference in Cell Flux Tally between Figure C.113 and Figure C.114 (using min/max of tally)

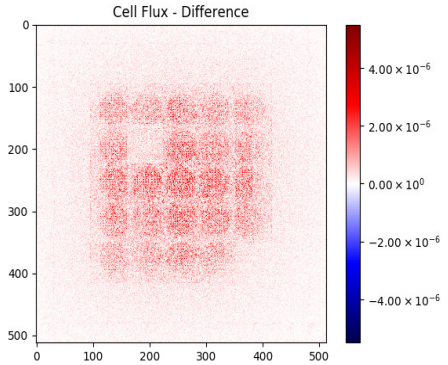


Figure C.116: Absolute Difference in Cell Flux Tally between Figure C.113 and Figure C.114 (using min/max of data)

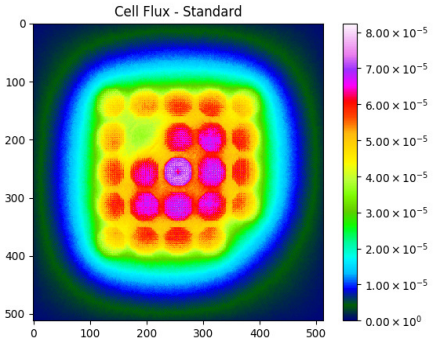


Figure C.117: Cell Flux Tally using standard tallies

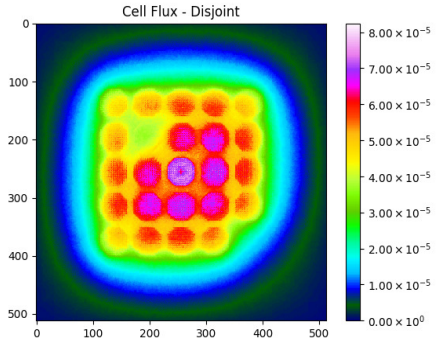


Figure C.118: Cell Flux Tally using disjoint tallies. $r_s = 0.75$

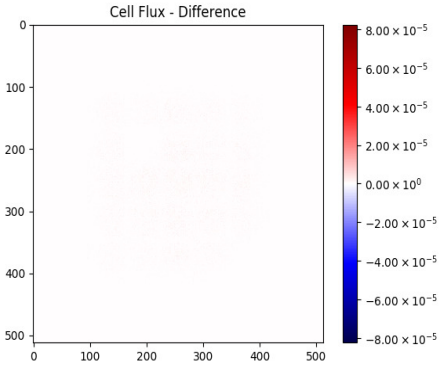


Figure C.119: Absolute Difference in Cell Flux Tally between Figure C.117 and Figure C.118 (using min/max of tally)

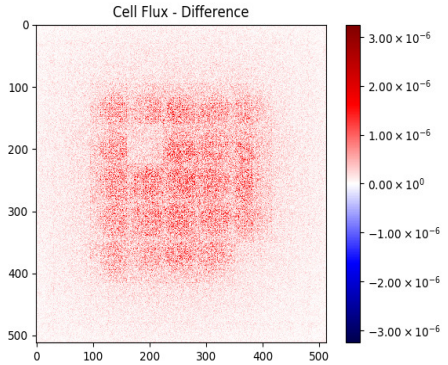


Figure C.120: Absolute Difference in Cell Flux Tally between Figure C.117 and Figure C.118 (using min/max of data)

C.2.3 Dose Deposit

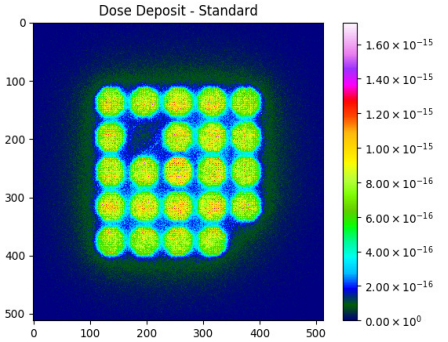


Figure C.121: Dose Deposit Tally using standard tallies

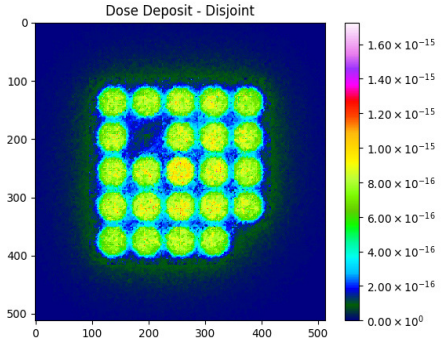


Figure C.122: Dose Deposit Tally using disjoint tallies. $r_s = 0.25$

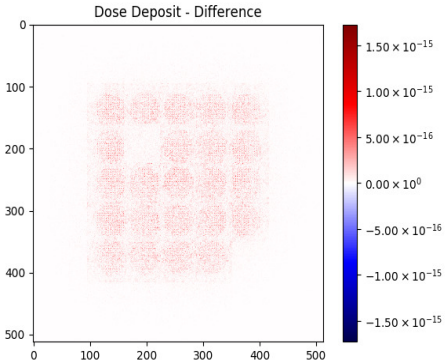


Figure C.123: Absolute Difference in Dose Deposit Tally between Figure C.121 and Figure C.122 (using min/max of tally)

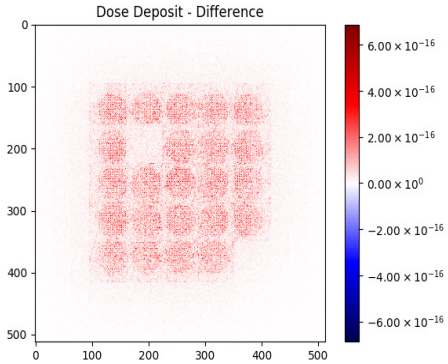


Figure C.124: Absolute Difference in Dose Deposit Tally between Figure C.121 and Figure C.122 (using min/max of data)

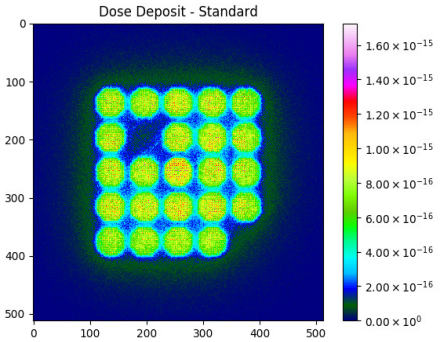


Figure C.125: Dose Deposit Tally using standard tallies

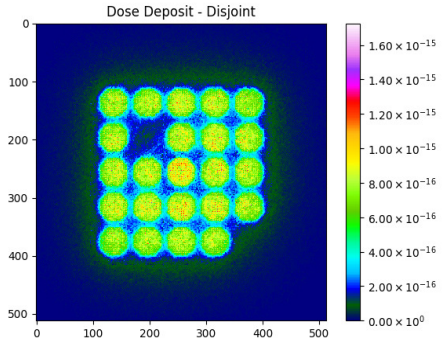


Figure C.126: Dose Deposit Tally using disjoint tallies. $r_s = 0.50$

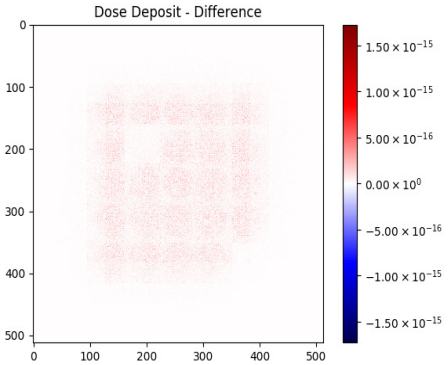


Figure C.127: Absolute Difference in Dose Deposit Tally between Figure C.125 and Figure C.126 (using min/max of tally)

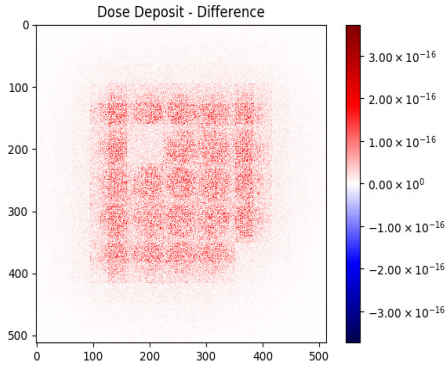


Figure C.128: Absolute Difference in Dose Deposit Tally between Figure C.125 and Figure C.126 (using min/max of data)

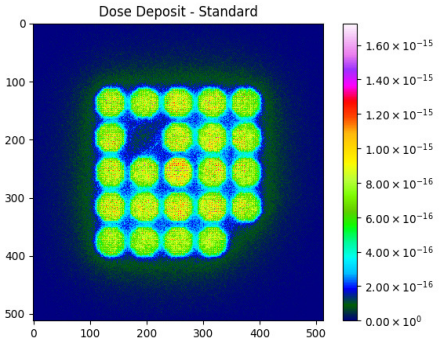


Figure C.129: Dose Deposit Tally using standard tallies

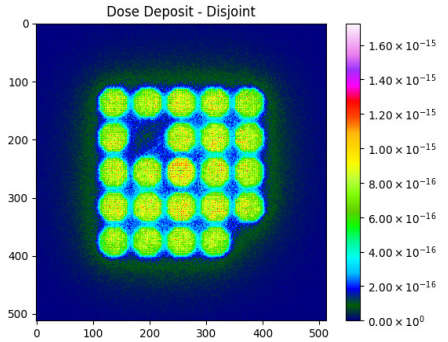


Figure C.130: Dose Deposit Tally using disjoint tallies. $r_s = 0.75$

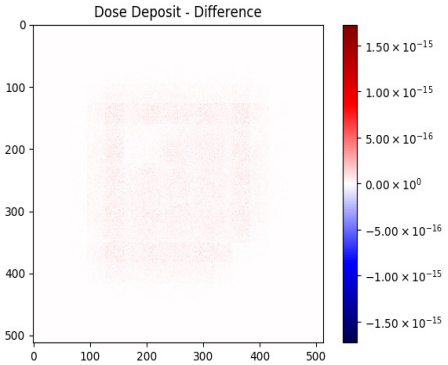


Figure C.131: Absolute Difference in Dose Deposit Tally between Figure C.129 and Figure C.130 (using min/max of tally)

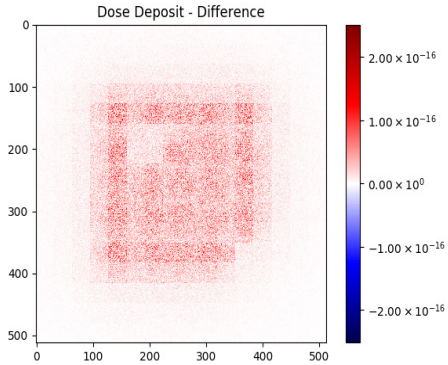


Figure C.132: Absolute Difference in Dose Deposit Tally between Figure C.129 and Figure C.130 (using min/max of data)

C.2.4 Energy Deposit

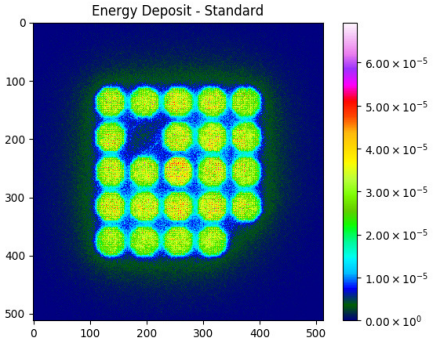


Figure C.133: Energy Deposit Tally using standard tallies

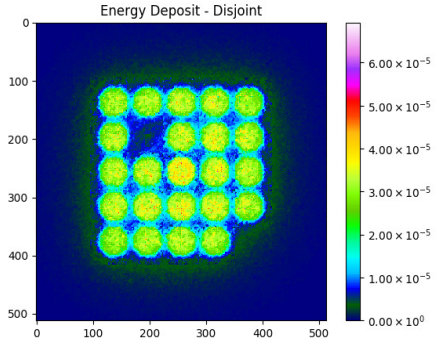


Figure C.134: Energy Deposit Tally using disjoint tallies. $r_s = 0.25$

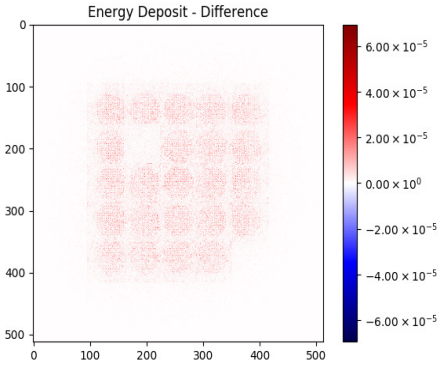


Figure C.135: Absolute Difference in Energy Deposit Tally between Figure C.133 and Figure C.134 (using min/max of tally)

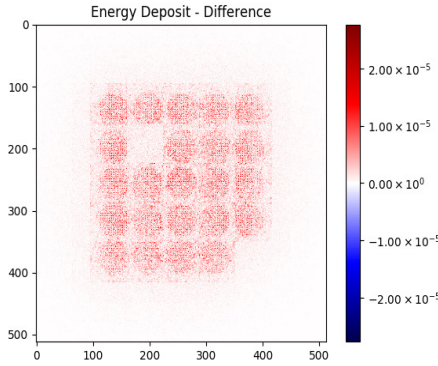


Figure C.136: Absolute Difference in Energy Deposit Tally between Figure C.133 and Figure C.134 (using min/max of data)

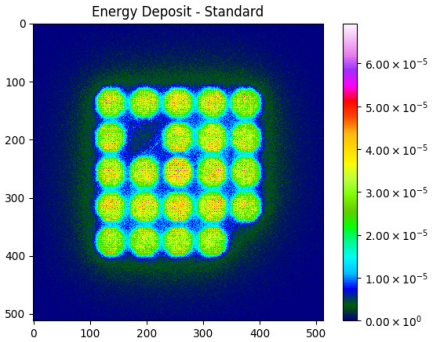


Figure C.137: Energy Deposit Tally using standard tallies

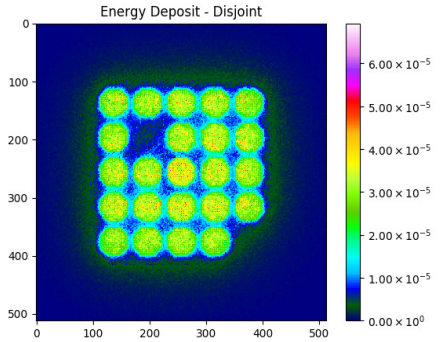


Figure C.138: Energy Deposit Tally using disjoint tallies. $r_s = 0.50$

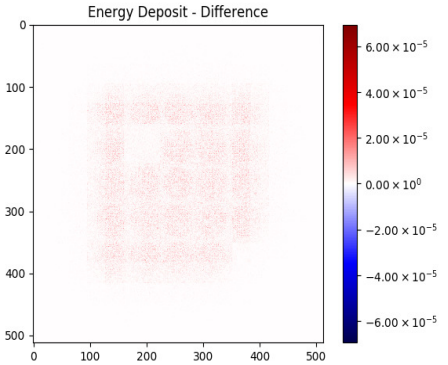


Figure C.139: Absolute Difference in Energy Deposit Tally between Figure C.137 and Figure C.138 (using min/max of tally)

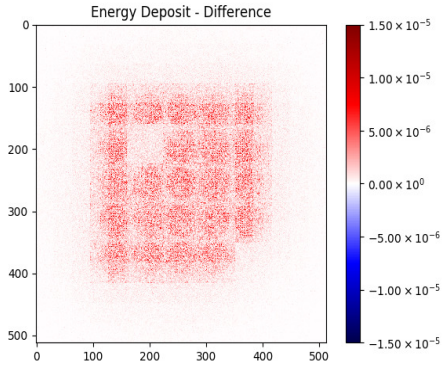


Figure C.140: Absolute Difference in Energy Deposit Tally between Figure C.137 and Figure C.138 (using min/max of data)

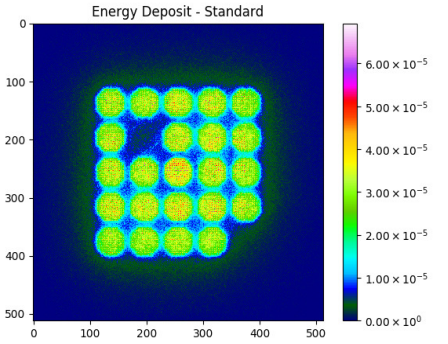


Figure C.141: Energy Deposit Tally using standard tallies

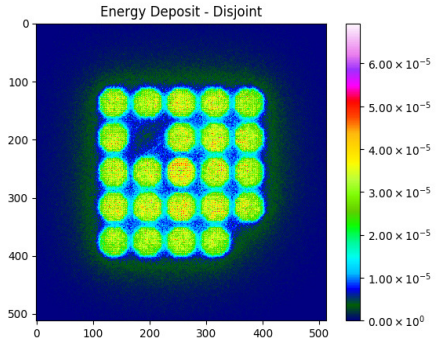


Figure C.142: Energy Deposit Tally using disjoint tallies. $r_s = 0.75$

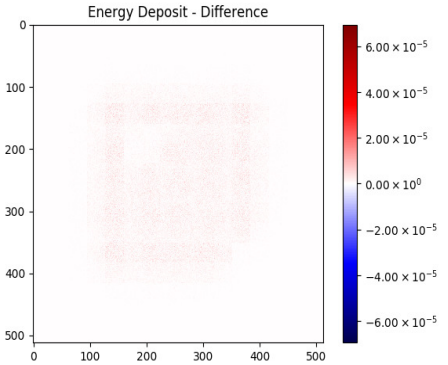


Figure C.143: Absolute Difference in Energy Deposit Tally between Figure C.141 and Figure C.142 (using min/max of tally)

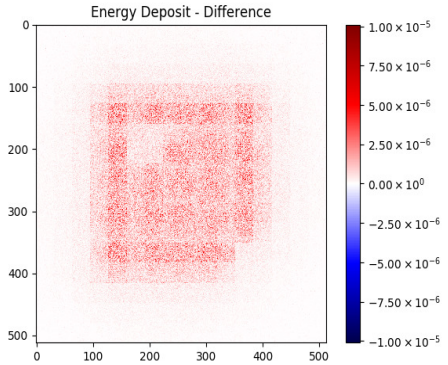


Figure C.144: Absolute Difference in Energy Deposit Tally between Figure C.141 and Figure C.142 (using min/max of data)

C.2.5 Flat Surface Current

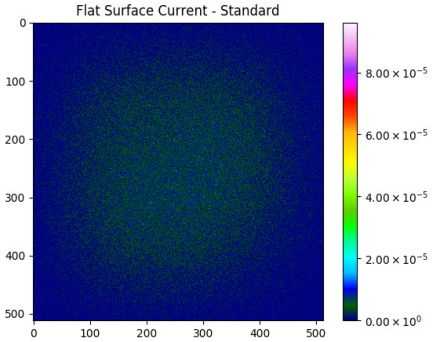


Figure C.145: Flat Surface Current Tally using standard tallies

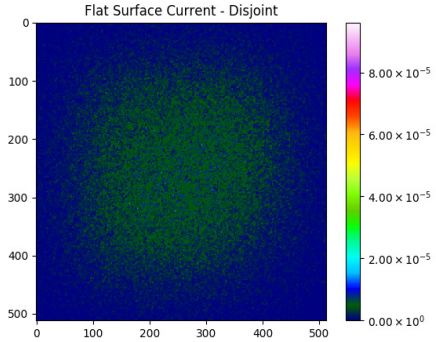


Figure C.146: Flat Surface Current Tally using disjoint tallies. $r_s = 0.25$

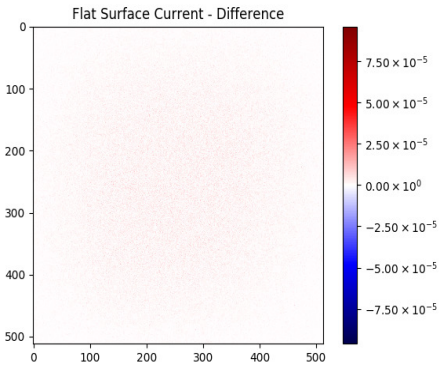


Figure C.147: Absolute Difference in Flat Surface Current Tally between Figure C.145 and Figure C.146 (using min/max of tally)

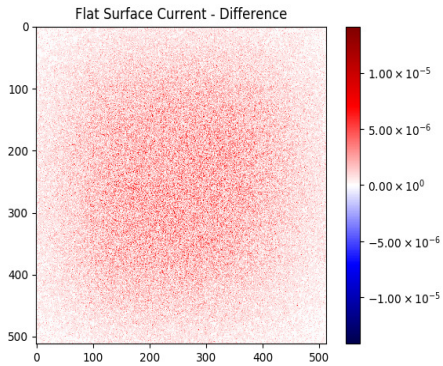


Figure C.148: Absolute Difference in Flat Surface Current Tally between Figure C.145 and Figure C.146 (using min/max of data)

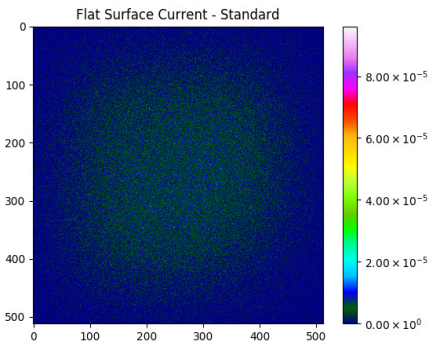


Figure C.149: Flat Surface Current Tally using standard tallies

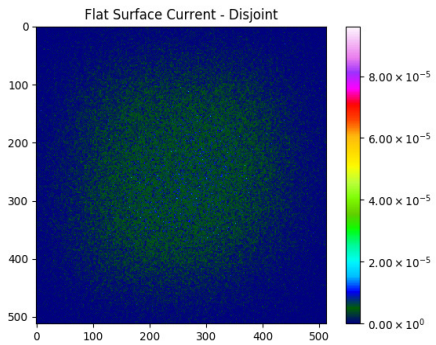


Figure C.150: Flat Surface Current Tally using disjoint tallies. $r_s = 0.50$

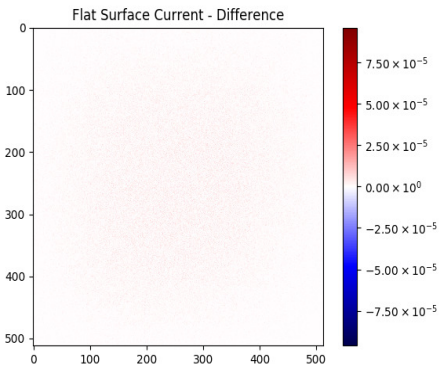


Figure C.151: Absolute Difference in Flat Surface Current Tally between Figure C.149 and Figure C.150 (using min/max of tally)

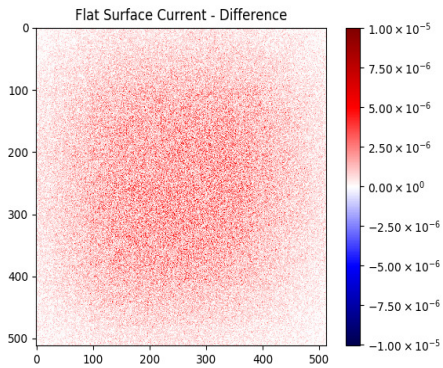


Figure C.152: Absolute Difference in Flat Surface Current Tally between Figure C.149 and Figure C.150 (using min/max of data)

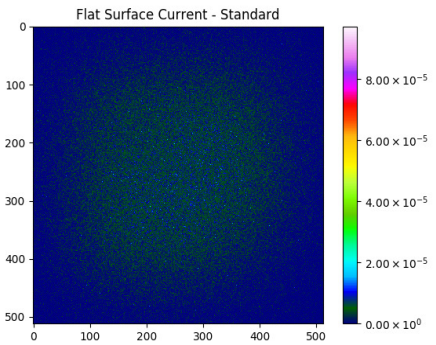


Figure C.153: Flat Surface Current Tally using standard tallies

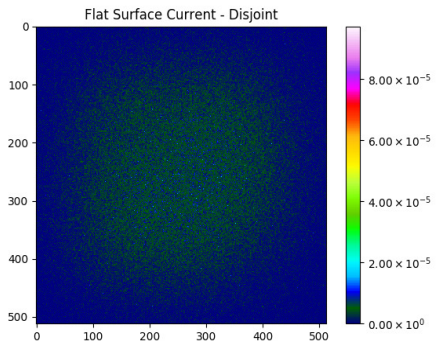


Figure C.154: Flat Surface Current Tally using disjoint tallies. $r_s = 0.75$

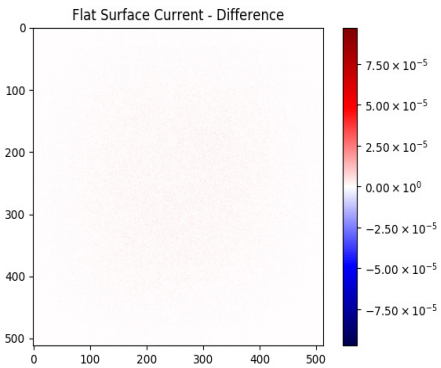


Figure C.155: Absolute Difference in Flat Surface Current Tally between Figure C.153 and Figure C.154 (using min/max of tally)

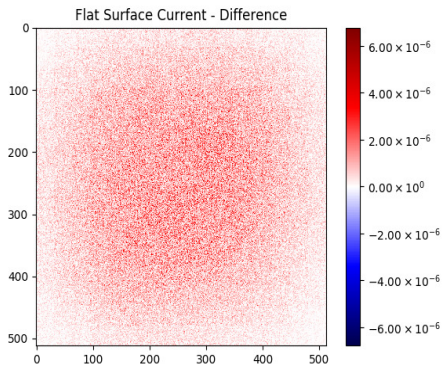


Figure C.156: Absolute Difference in Flat Surface Current Tally between Figure C.153 and Figure C.154 (using min/max of data)

C.2.6 Number of Collision

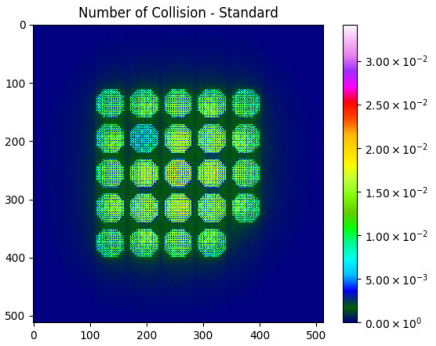


Figure C.157: Number of Collision Tally using standard tallies

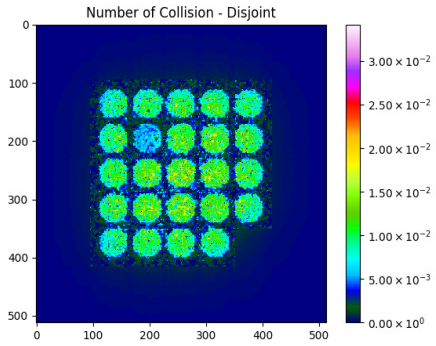


Figure C.158: Number of Collision Tally using disjoint tallies. $r_s = 0.25$

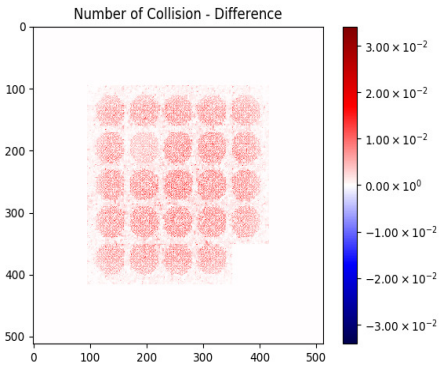


Figure C.159: Absolute Difference in Number of Collision Tally between Figure C.157 and Figure C.158 (using min/max of tally)

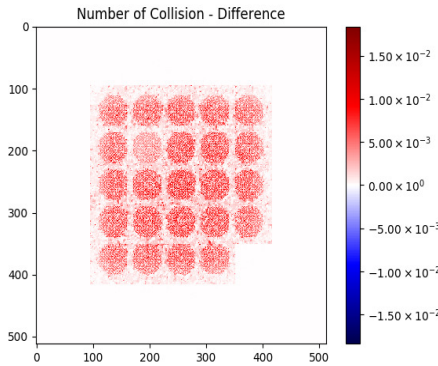


Figure C.160: Absolute Difference in Number of Collision Tally between Figure C.157 and Figure C.158 (using min/max of data)

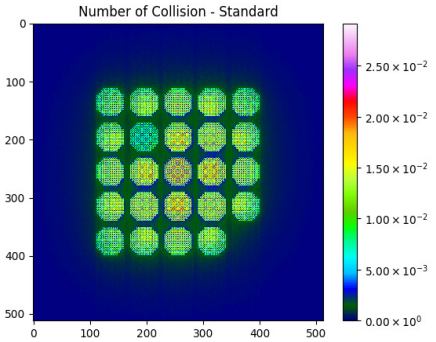


Figure C.161: Number of Collision Tally using standard tallies

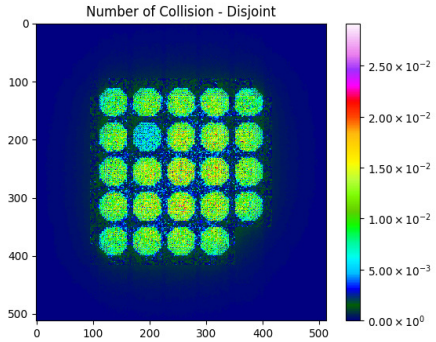


Figure C.162: Number of Collision Tally using disjoint tallies. $r_s = 0.50$

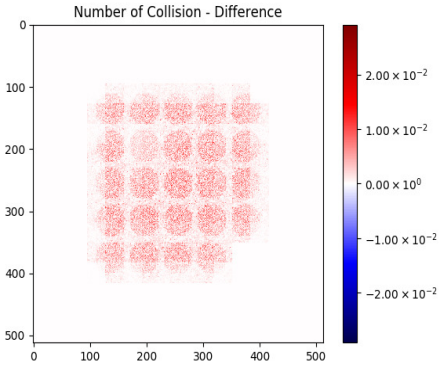


Figure C.163: Absolute Difference in Number of Collision Tally between Figure C.161 and Figure C.162 (using min/max of tally)

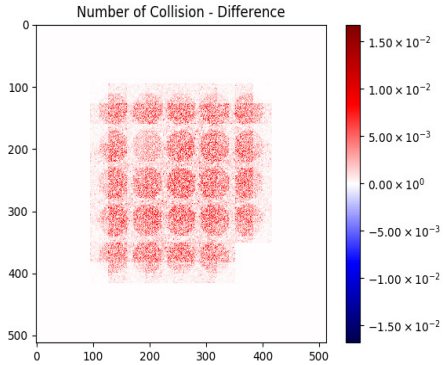


Figure C.164: Absolute Difference in Number of Collision Tally between Figure C.161 and Figure C.162 (using min/max of data)

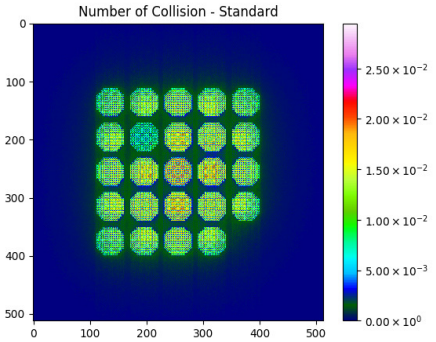


Figure C.165: Number of Collision Tally using standard tallies

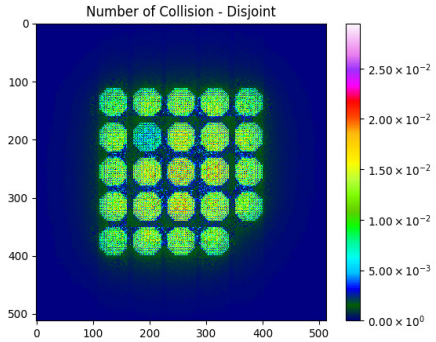


Figure C.166: Number of Collision Tally using disjoint tallies. $r_s = 0.75$

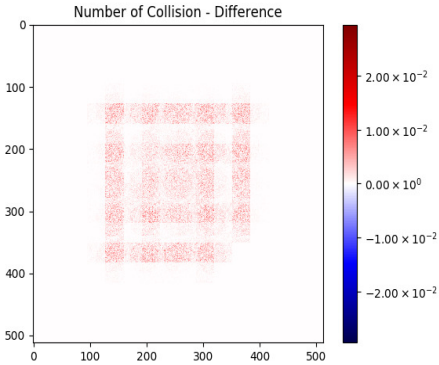


Figure C.167: Absolute Difference in Number of Collision Tally between Figure C.165 and Figure C.166 (using min/max of tally)

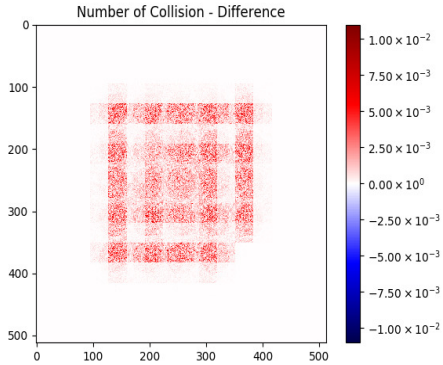


Figure C.168: Absolute Difference in Number of Collision Tally between Figure C.165 and Figure C.166 (using min/max of data)

C.2.7 Number of Secondary

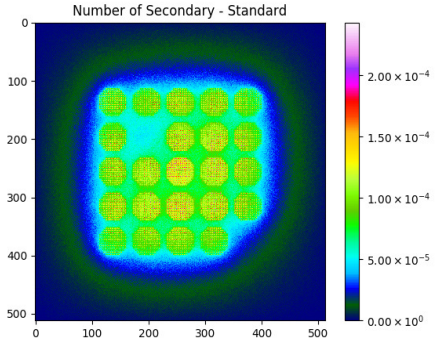


Figure C.169: Number of Secondary Tally using standard tallies

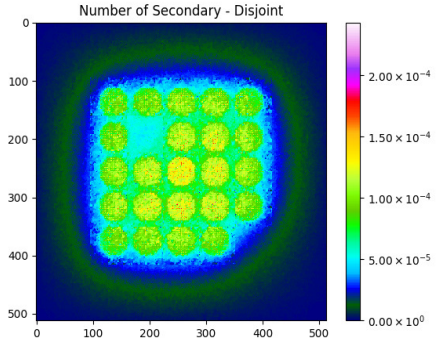


Figure C.170: Number of Secondary Tally using disjoint tallies. $r_s = 0.25$

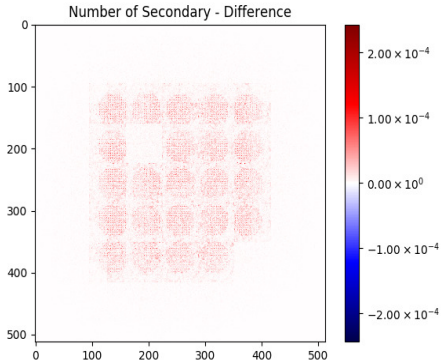


Figure C.171: Absolute Difference in Number of Secondary Tally between Figure C.169 and Figure C.170 (using min/max of tally)

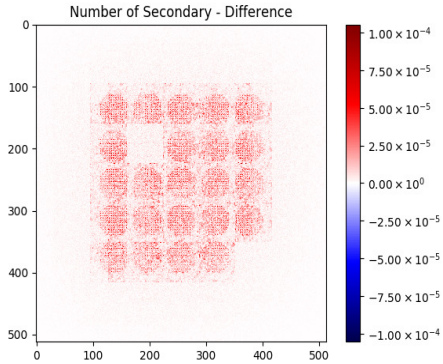


Figure C.172: Absolute Difference in Number of Secondary Tally between Figure C.169 and Figure C.170 (using min/max of data)

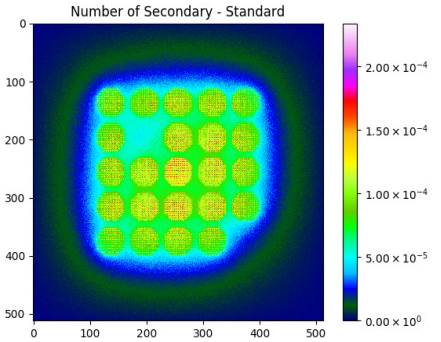


Figure C.173: Number of Secondary Tally using standard tallies

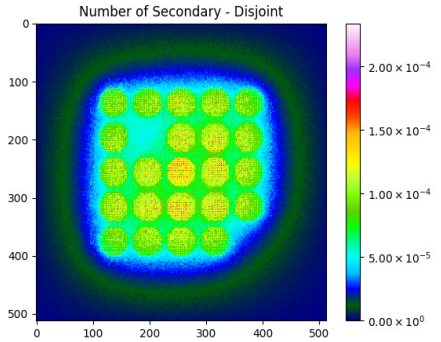


Figure C.174: Number of Secondary Tally using disjoint tallies. $r_s = 0.50$

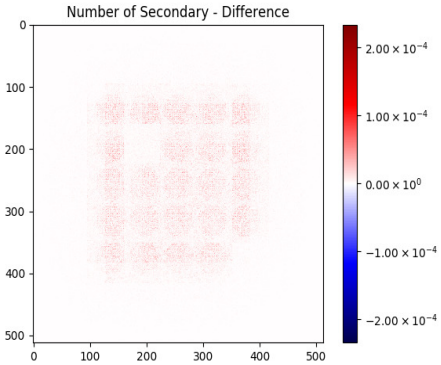


Figure C.175: Absolute Difference in Number of Secondary Tally between Figure C.173 and Figure C.174 (using min/max of tally)

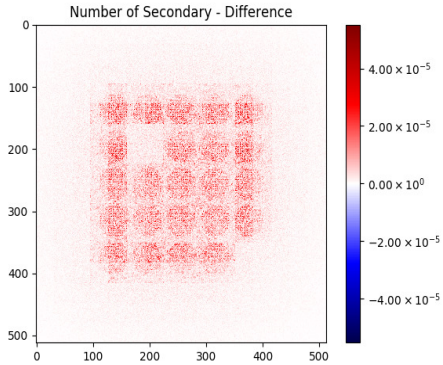


Figure C.176: Absolute Difference in Number of Secondary Tally between Figure C.173 and Figure C.174 (using min/max of data)

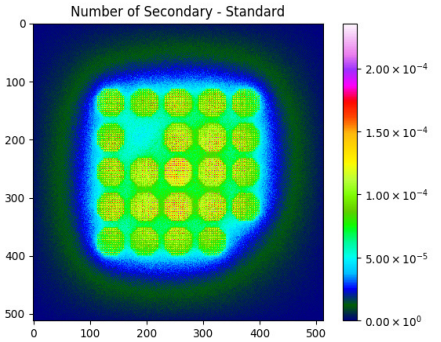


Figure C.177: Number of Secondary Tally using standard tallies

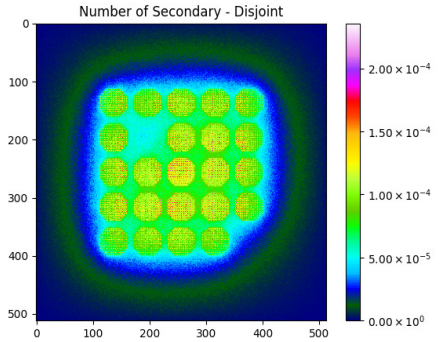


Figure C.178: Number of Secondary Tally using disjoint tallies. $r_s = 0.75$

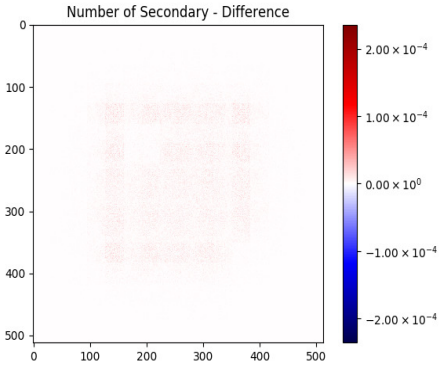


Figure C.179: Absolute Difference in Number of Secondary Tally between Figure C.177 and Figure C.178 (using min/max of tally)

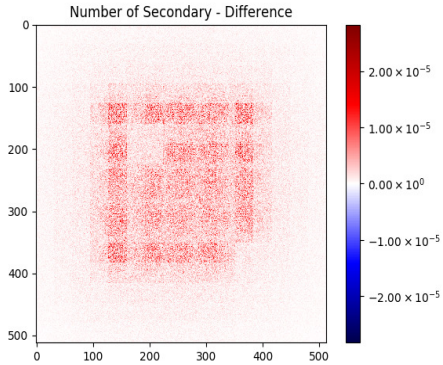


Figure C.180: Absolute Difference in Number of Secondary Tally between Figure C.177 and Figure C.178 (using min/max of data)

C.2.8 Number of Step

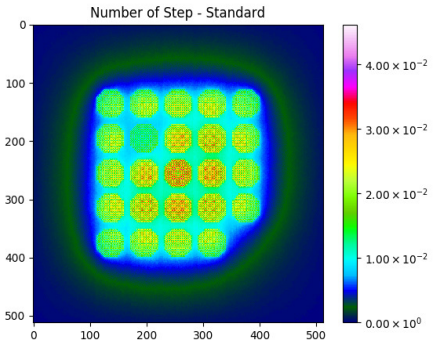


Figure C.181: Number of Step Tally using standard tallies

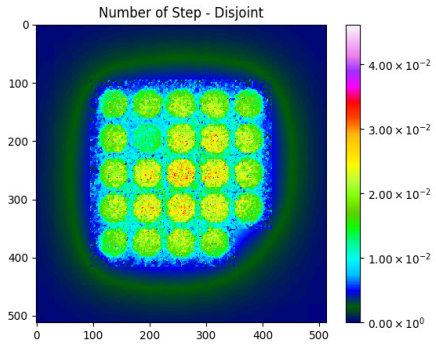


Figure C.182: Number of Step Tally using disjoint tallies. $r_s = 0.25$

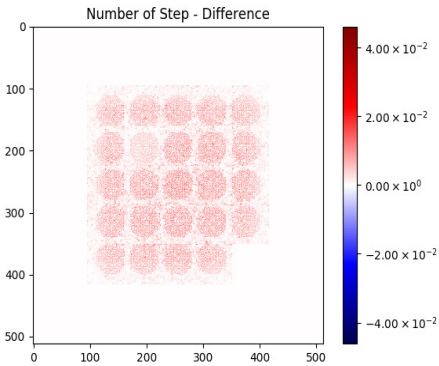


Figure C.183: Absolute Difference in Number of Step Tally between Figure C.181 and Figure C.182 (using min/max of tally)

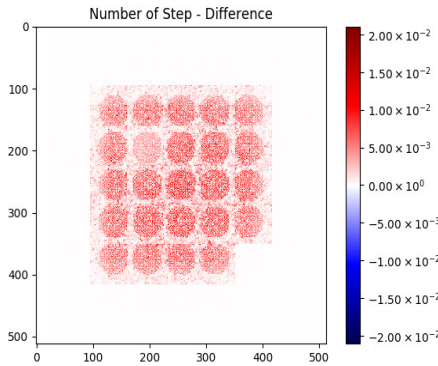


Figure C.184: Absolute Difference in Number of Step Tally between Figure C.181 and Figure C.182 (using min/max of data)

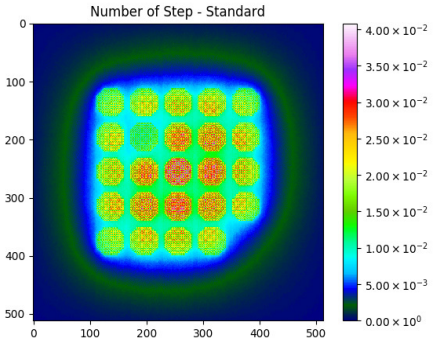


Figure C.185: Number of Step Tally using standard tallies

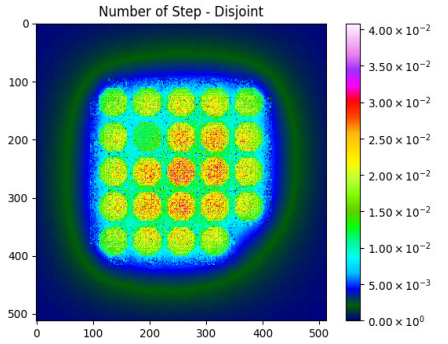


Figure C.186: Number of Step Tally using disjoint tallies. $r_s = 0.50$

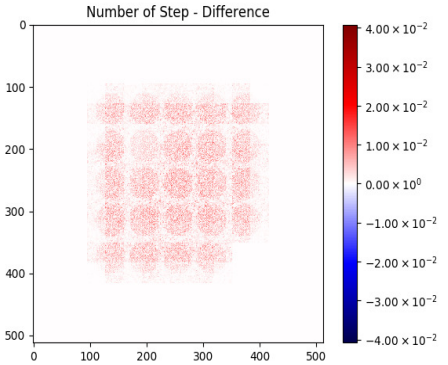


Figure C.187: Absolute Difference in Number of Step Tally between Figure C.185 and Figure C.186 (using min/max of tally)

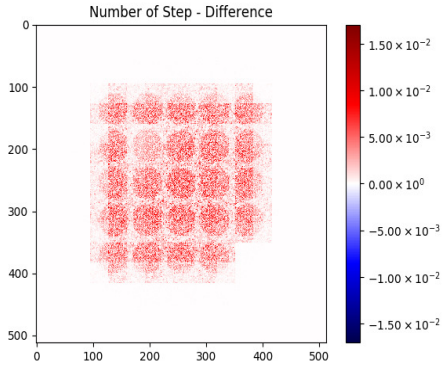


Figure C.188: Absolute Difference in Number of Step Tally between Figure C.185 and Figure C.186 (using min/max of data)

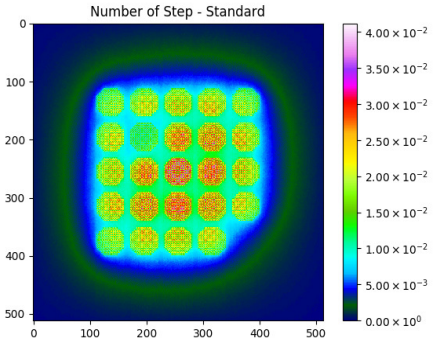


Figure C.189: Number of Step Tally using standard tallies

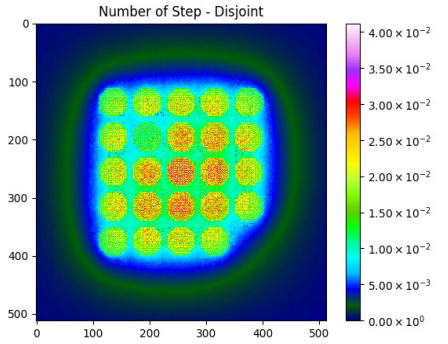


Figure C.190: Number of Step Tally using disjoint tallies. $r_s = 0.75$

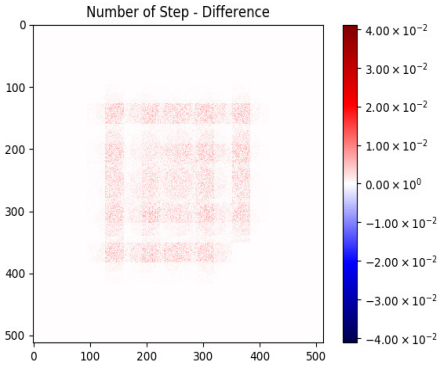


Figure C.191: Absolute Difference in Number of Step Tally between Figure C.189 and Figure C.190 (using min/max of tally)

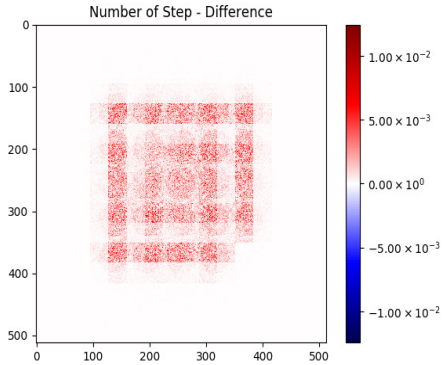


Figure C.192: Absolute Difference in Number of Step Tally between Figure C.189 and Figure C.190 (using min/max of data)

C.2.9 Passage Cell Current

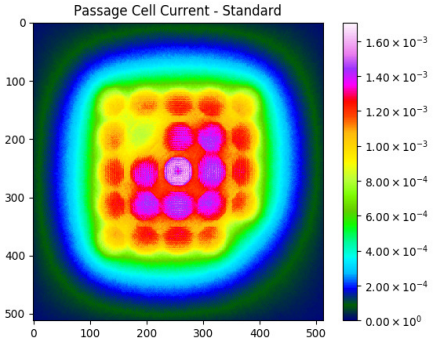


Figure C.193: Passage Cell Current Tally using standard tallies

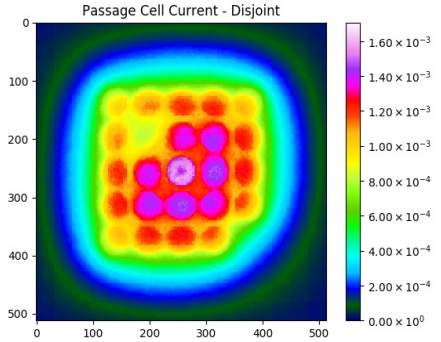


Figure C.194: Passage Cell Current Tally using disjoint tallies. $r_s = 0.25$

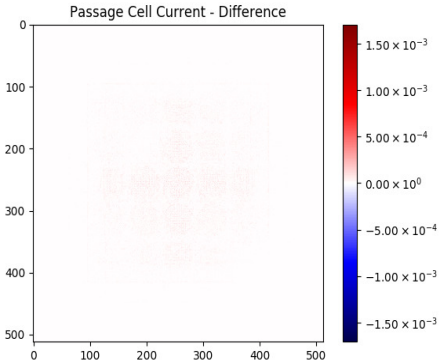


Figure C.195: Absolute Difference in Passage Cell Current Tally between Figure C.193 and Figure C.194 (using min/max of tally)

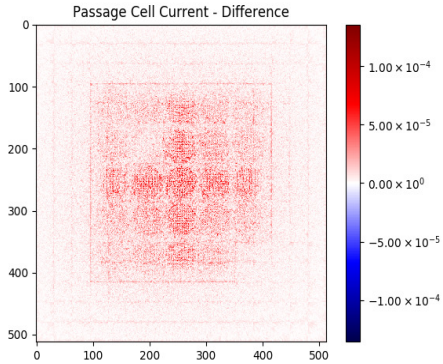


Figure C.196: Absolute Difference in Passage Cell Current Tally between Figure C.193 and Figure C.194 (using min/max of data)

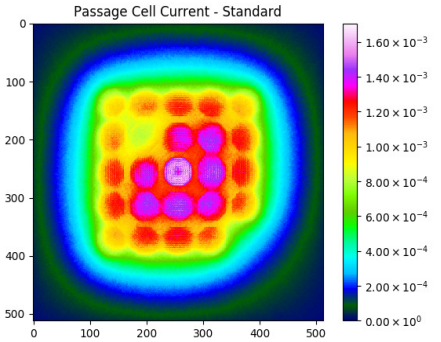


Figure C.197: Passage Cell Current Tally using standard tallies

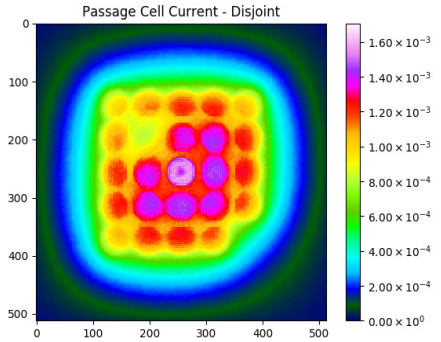


Figure C.198: Passage Cell Current Tally using disjoint tallies. $r_s = 0.50$

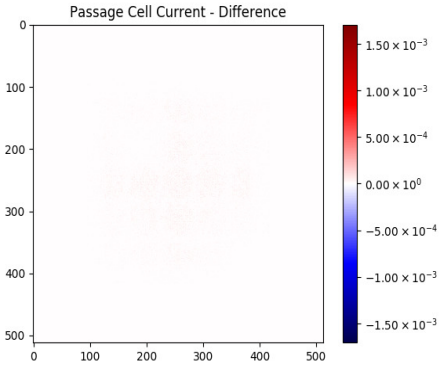


Figure C.199: Absolute Difference in Passage Cell Current Tally between Figure C.197 and Figure C.198 (using min/max of tally)

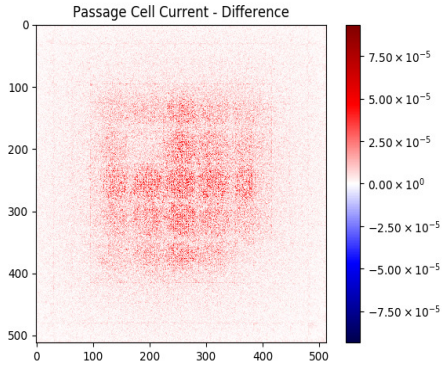


Figure C.200: Absolute Difference in Passage Cell Current Tally between Figure C.197 and Figure C.198 (using min/max of data)

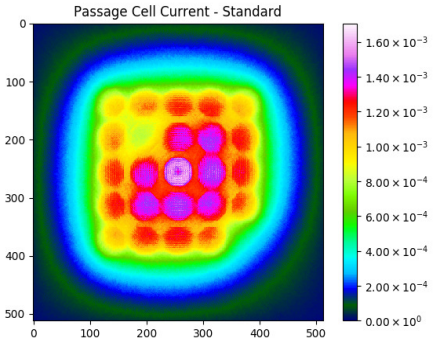


Figure C.201: Passage Cell Current Tally using standard tallies

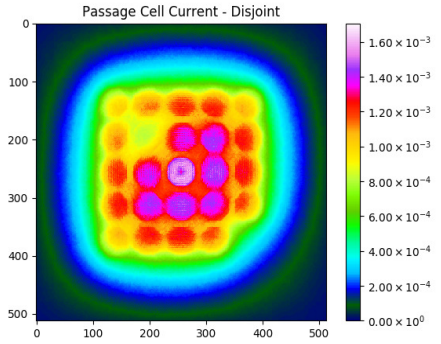


Figure C.202: Passage Cell Current Tally using disjoint tallies. $r_s = 0.75$

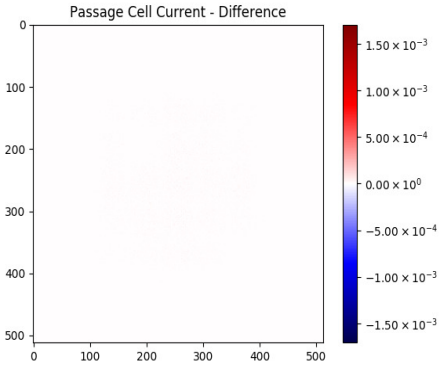


Figure C.203: Absolute Difference in Passage Cell Current Tally between Figure C.201 and Figure C.202 (using min/max of tally)

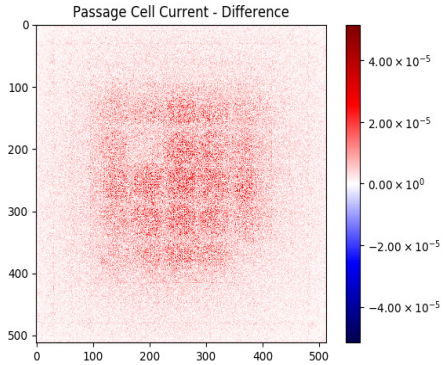


Figure C.204: Absolute Difference in Passage Cell Current Tally between Figure C.201 and Figure C.202 (using min/max of data)

C.2.10 Passage Cell Flux

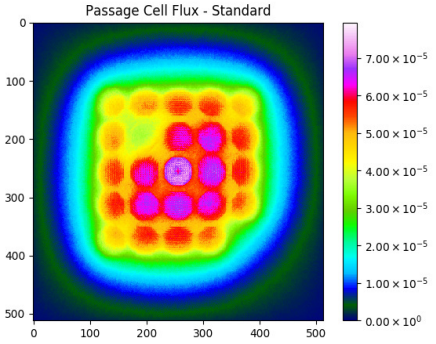


Figure C.205: Passage Cell Flux Tally using standard tallies

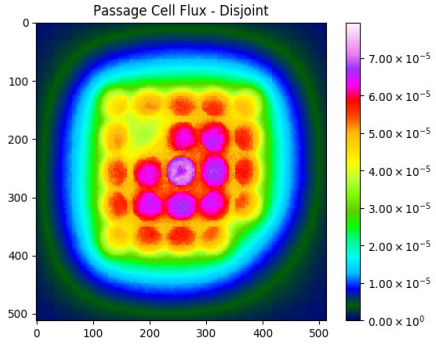


Figure C.206: Passage Cell Flux Tally using disjoint tallies. $r_s = 0.25$

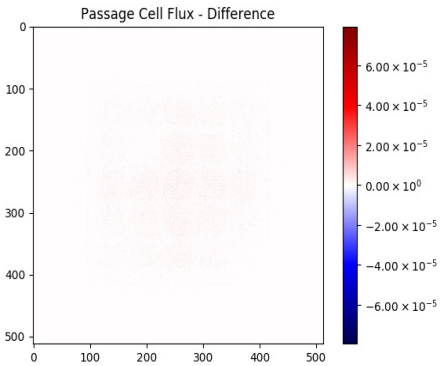


Figure C.207: Absolute Difference in Passage Cell Flux Tally between Figure C.205 and Figure C.206 (using min/max of tally)

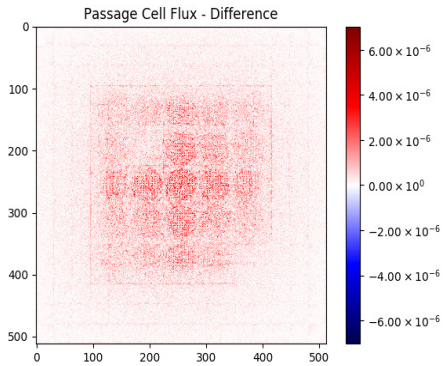


Figure C.208: Absolute Difference in Passage Cell Flux Tally between Figure C.205 and Figure C.206 (using min/max of data)

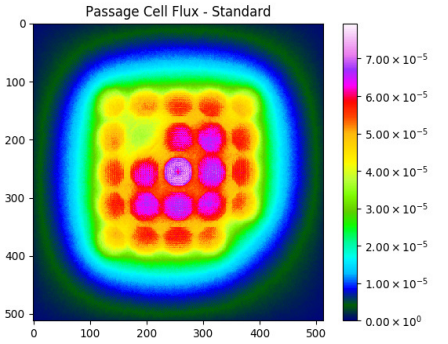


Figure C.209: Passage Cell Flux Tally using standard tallies

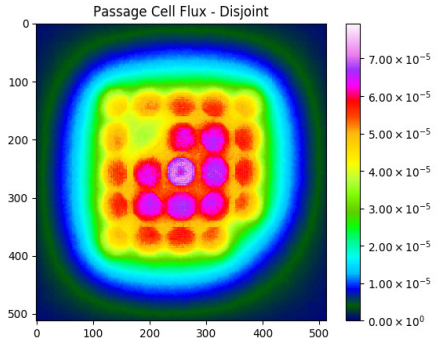


Figure C.210: Passage Cell Flux Tally using disjoint tallies. $r_s = 0.50$

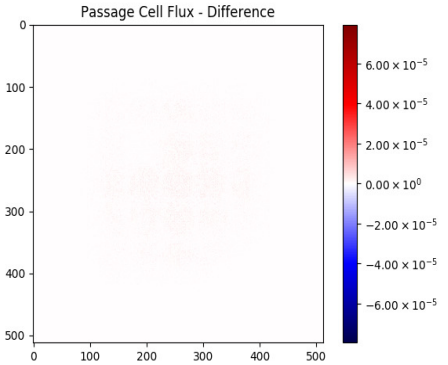


Figure C.211: Absolute Difference in Passage Cell Flux Tally between Figure C.209 and Figure C.210 (using min/max of tally)

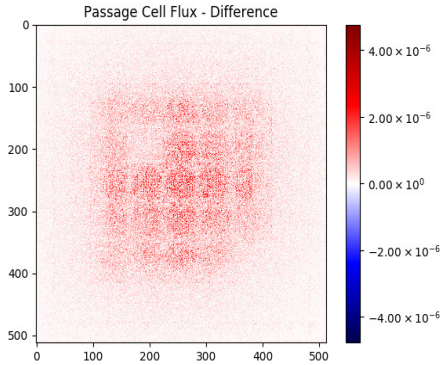


Figure C.212: Absolute Difference in Passage Cell Flux Tally between Figure C.209 and Figure C.210 (using min/max of data)

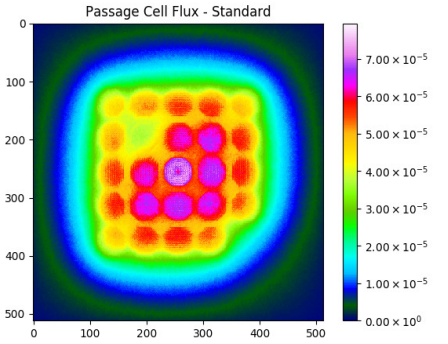


Figure C.213: Passage Cell Flux Tally using standard tallies

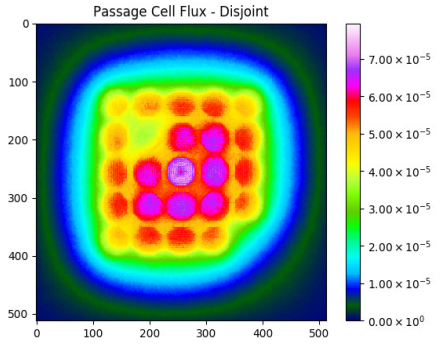


Figure C.214: Passage Cell Flux Tally using disjoint tallies. $r_s = 0.75$

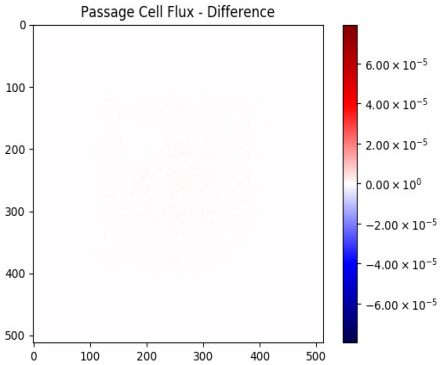


Figure C.215: Absolute Difference in Passage Cell Flux Tally between Figure C.213 and Figure C.214 (using min/max of tally)

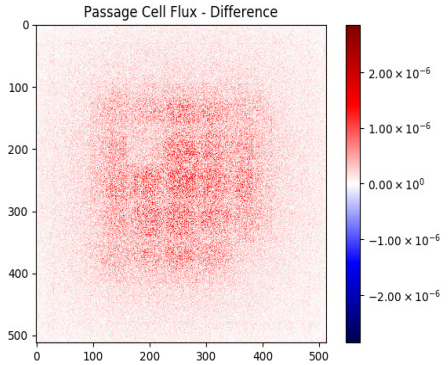


Figure C.216: Absolute Difference in Passage Cell Flux Tally between Figure C.213 and Figure C.214 (using min/max of data)

C.2.11 Passage Track Length

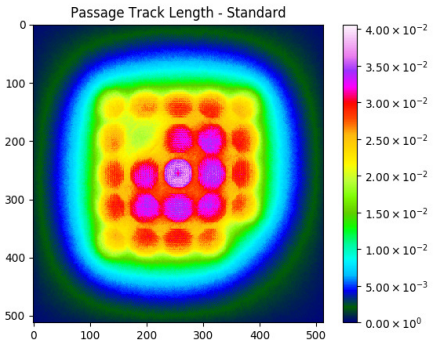


Figure C.217: Passage Track Length Tally using standard tallies

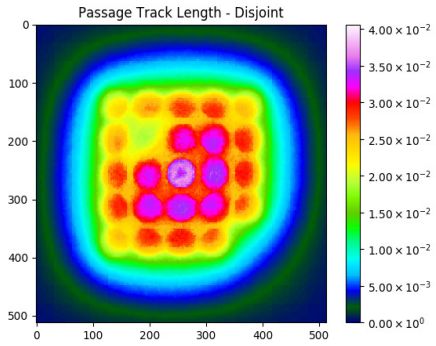


Figure C.218: Passage Track Length Tally using disjoint tallies. $r_s = 0.25$

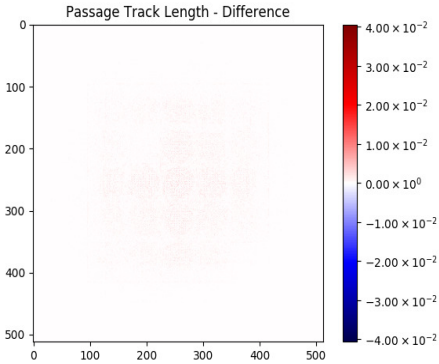


Figure C.219: Absolute Difference in Passage Track Length Tally between Figure C.217 and Figure C.218 (using min/max of tally)

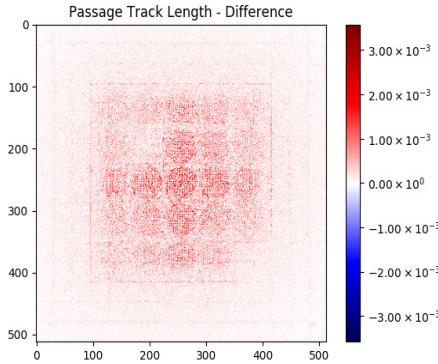


Figure C.220: Absolute Difference in Passage Track Length Tally between Figure C.217 and Figure C.218 (using min/max of data)

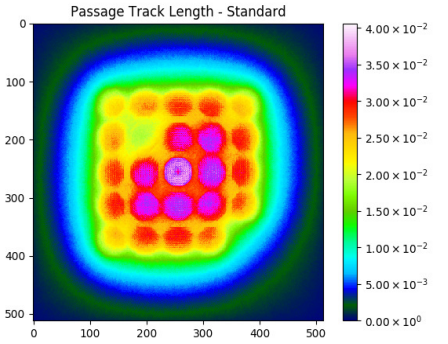


Figure C.221: Passage Track Length Tally using standard tallies

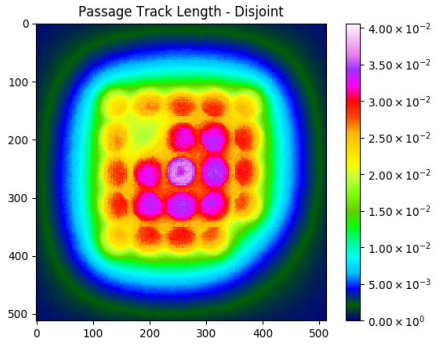


Figure C.222: Passage Track Length Tally using disjoint tallies. $r_s = 0.50$

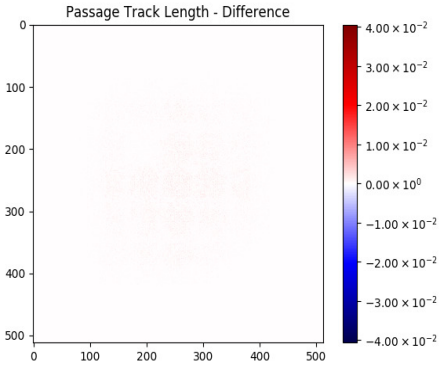


Figure C.223: Absolute Difference in Passage Track Length Tally between Figure C.221 and Figure C.222 (using min/max of tally)

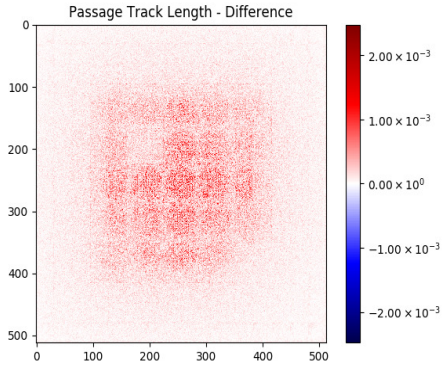


Figure C.224: Absolute Difference in Passage Track Length Tally between Figure C.221 and Figure C.222 (using min/max of data)

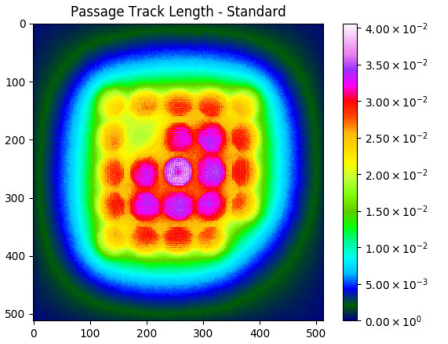


Figure C.225: Passage Track Length Tally using standard tallies

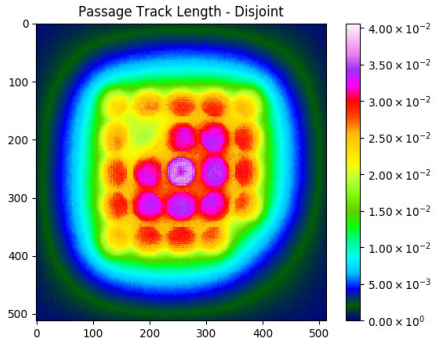


Figure C.226: Passage Track Length Tally using disjoint tallies. $r_s = 0.75$

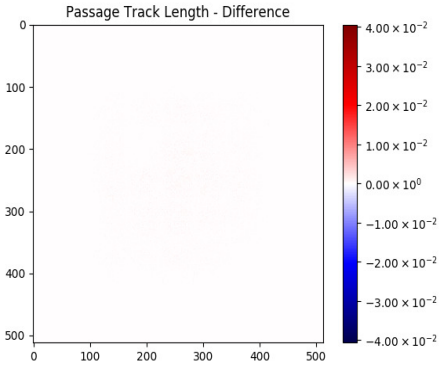


Figure C.227: Absolute Difference in Passage Track Length Tally between Figure C.225 and Figure C.226 (using min/max of tally)

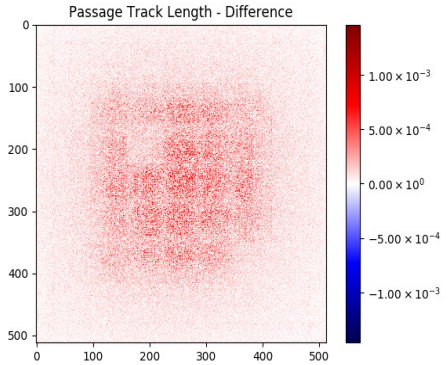


Figure C.228: Absolute Difference in Passage Track Length Tally between Figure C.225 and Figure C.226 (using min/max of data)

C.2.12 Population

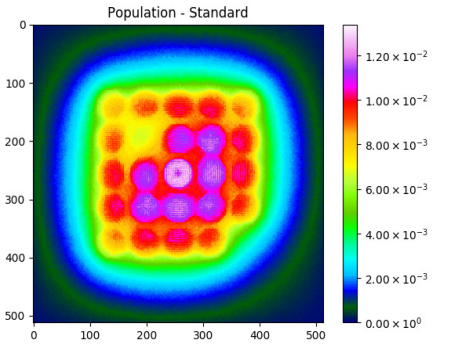


Figure C.229: Population Tally using standard tallies

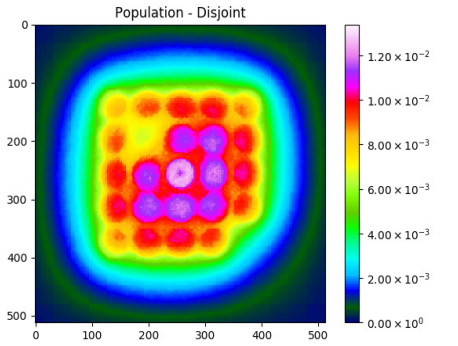


Figure C.230: Population Tally using disjoint tallies. $r_s = 0.25$

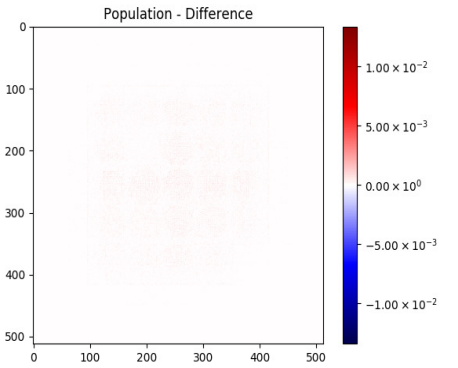


Figure C.231: Absolute Difference in Population Tally between Figure C.229 and Figure C.230 (using min/max of tally)

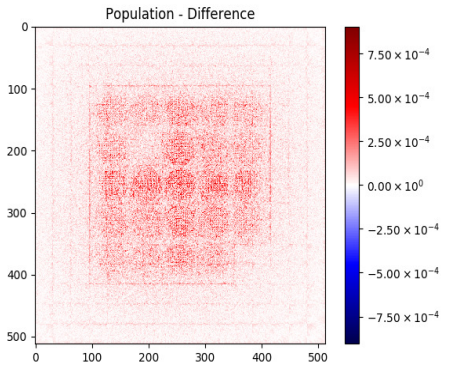


Figure C.232: Absolute Difference in Population Tally between Figure C.229 and Figure C.230 (using min/max of data)

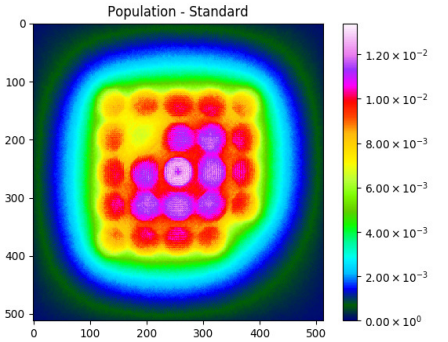


Figure C.233: Population Tally using standard tallies

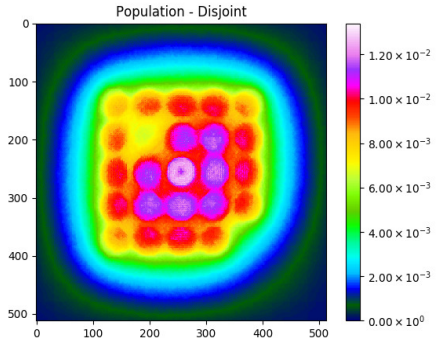


Figure C.234: Population Tally using disjoint tallies. $r_s = 0.50$

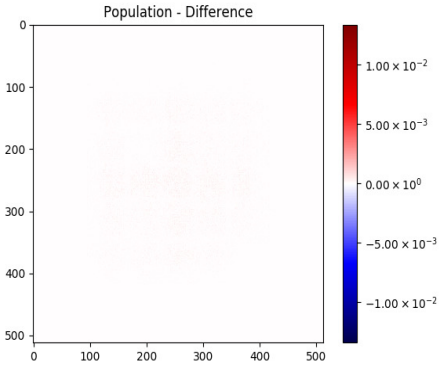


Figure C.235: Absolute Difference in Population Tally between Figure C.233 and Figure C.234 (using min/max of tally)

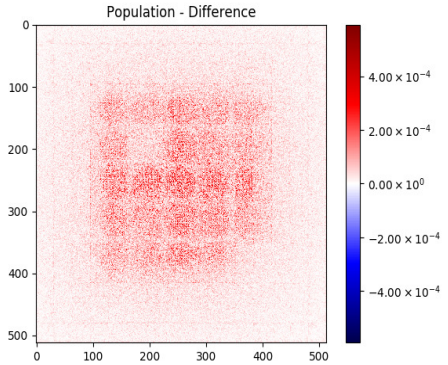


Figure C.236: Absolute Difference in Population Tally between Figure C.233 and Figure C.234 (using min/max of data)

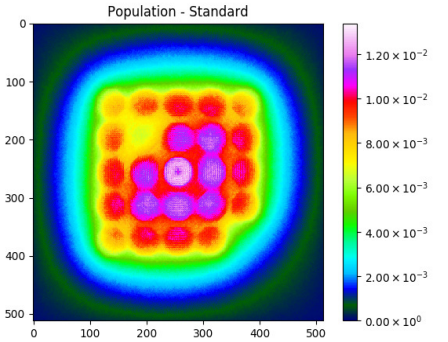


Figure C.237: Population Tally using standard tallies

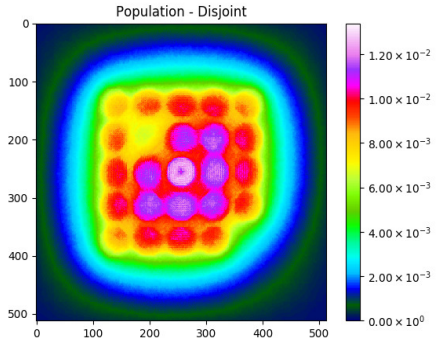


Figure C.238: Population Tally using disjoint tallies. $r_s = 0.75$

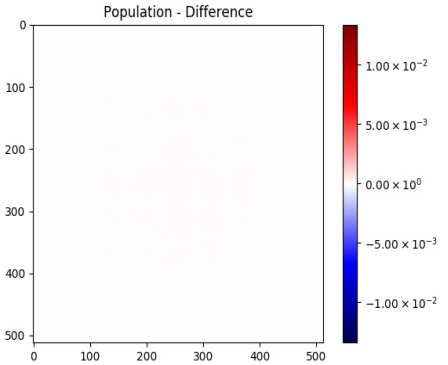


Figure C.239: Absolute Difference in Population Tally between Figure C.237 and Figure C.238 (using min/max of tally)

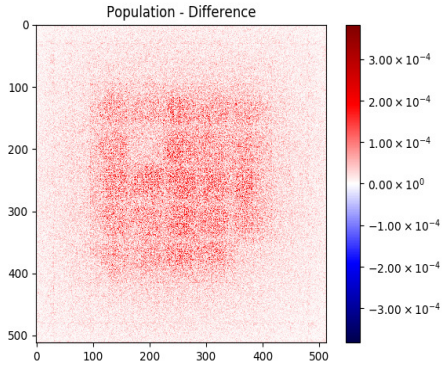


Figure C.240: Absolute Difference in Population Tally between Figure C.237 and Figure C.238 (using min/max of data)

C.2.13 Termination

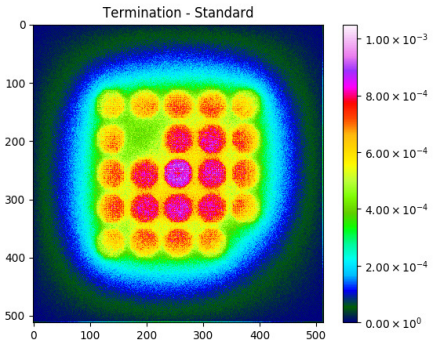


Figure C.241: Termination Tally using standard tallies

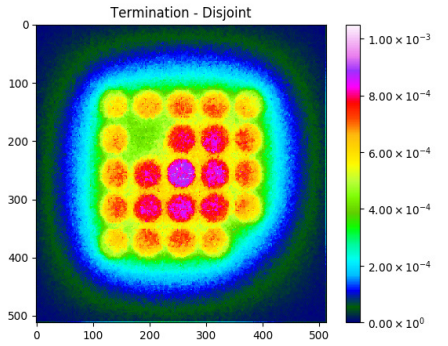


Figure C.242: Termination Tally using disjoint tallies. $r_s = 0.25$

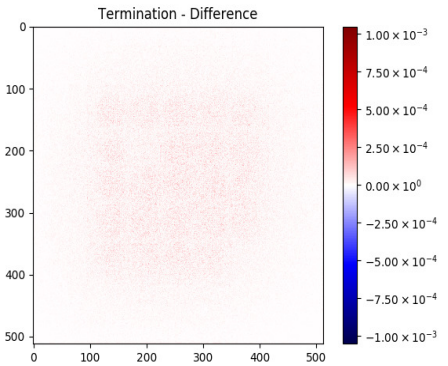


Figure C.243: Absolute Difference in Termination Tally between Figure C.241 and Figure C.242 (using min/max of tally)

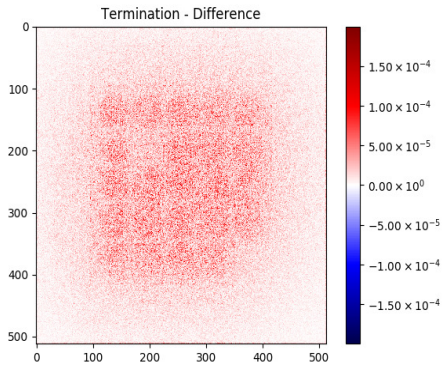


Figure C.244: Absolute Difference in Termination Tally between Figure C.241 and Figure C.242 (using min/max of data)

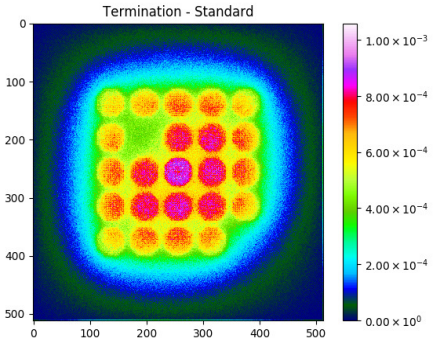


Figure C.245: Termination Tally using standard tallies

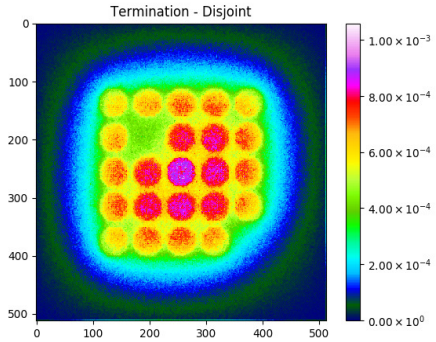


Figure C.246: Termination Tally using disjoint tallies. $r_s = 0.50$

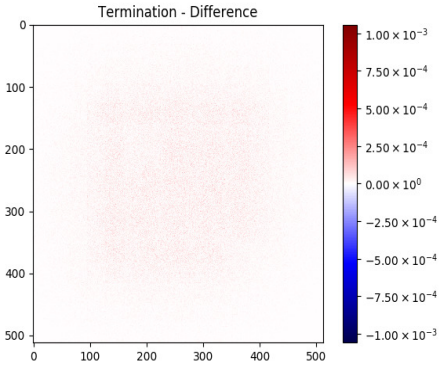


Figure C.247: Absolute Difference in Termination Tally between Figure C.245 and Figure C.246 (using min/max of tally)

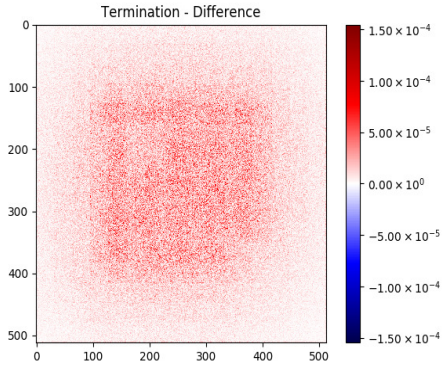


Figure C.248: Absolute Difference in Termination Tally between Figure C.245 and Figure C.246 (using min/max of data)

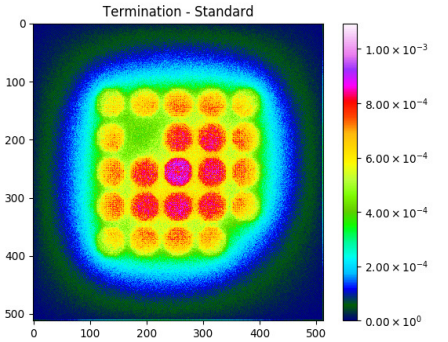


Figure C.249: Termination Tally using standard tallies

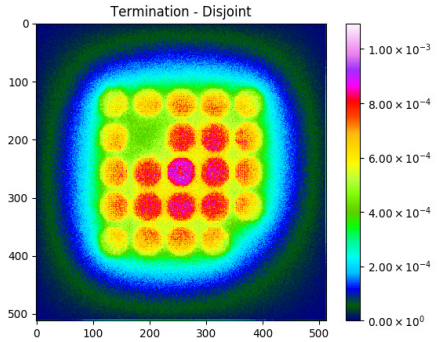


Figure C.250: Termination Tally using disjoint tallies. $r_s = 0.75$

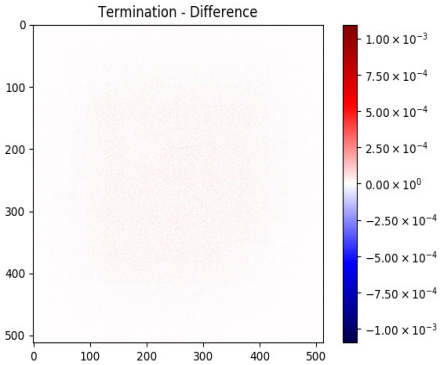


Figure C.251: Absolute Difference in Termination Tally between Figure C.249 and Figure C.250 (using min/max of tally)

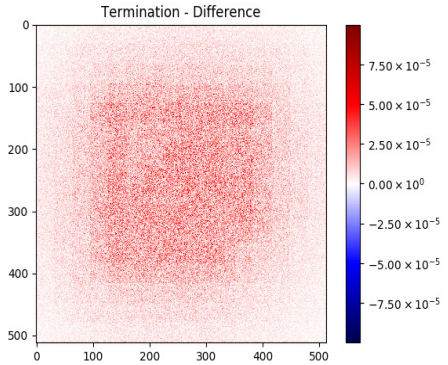


Figure C.252: Absolute Difference in Termination Tally between Figure C.249 and Figure C.250 (using min/max of data)

C.2.14 Track Counter

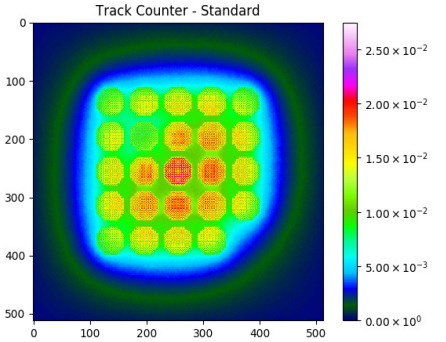


Figure C.253: Track Counter Tally using standard tallies

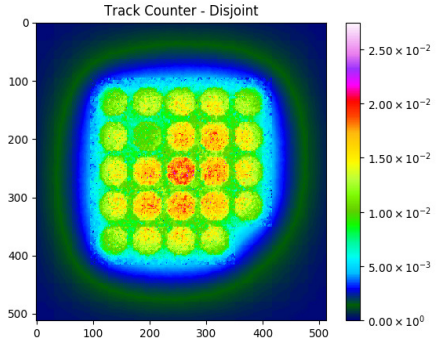


Figure C.254: Track Counter Tally using disjoint tallies. $r_s = 0.25$

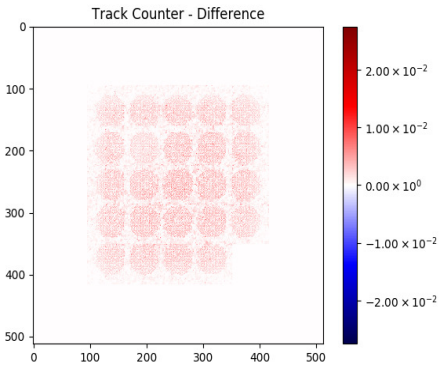


Figure C.255: Absolute Difference in Track Counter Tally between Figure C.253 and Figure C.254 (using min/max of tally)

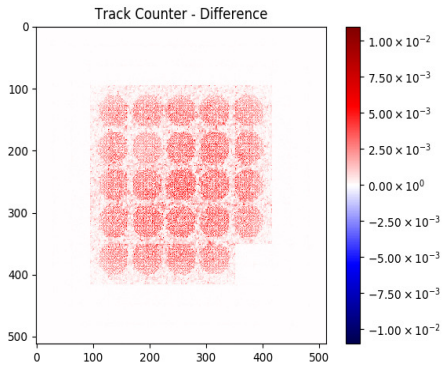


Figure C.256: Absolute Difference in Track Counter Tally between Figure C.253 and Figure C.254 (using min/max of data)

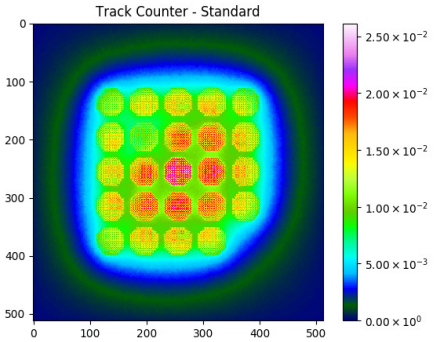


Figure C.257: Track Counter Tally using standard tallies

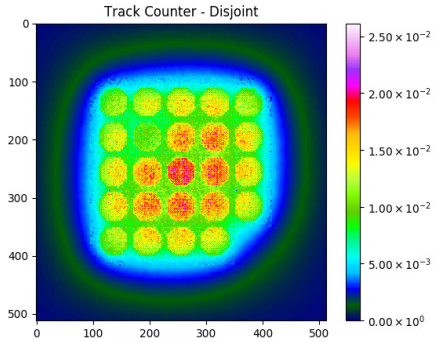


Figure C.258: Track Counter Tally using disjoint tallies. $r_s = 0.50$

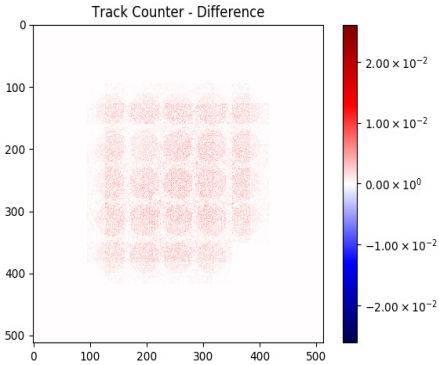


Figure C.259: Absolute Difference in Track Counter Tally between Figure C.257 and Figure C.258 (using min/max of tally)

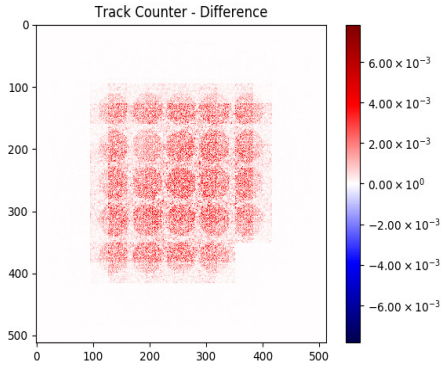


Figure C.260: Absolute Difference in Track Counter Tally between Figure C.257 and Figure C.258 (using min/max of data)

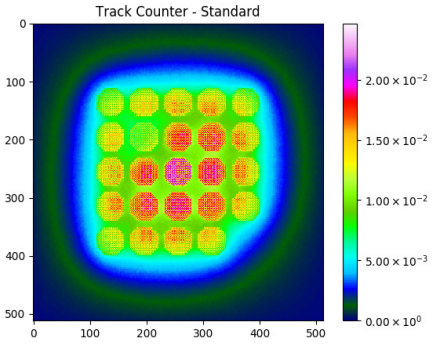


Figure C.261: Track Counter Tally using standard tallies

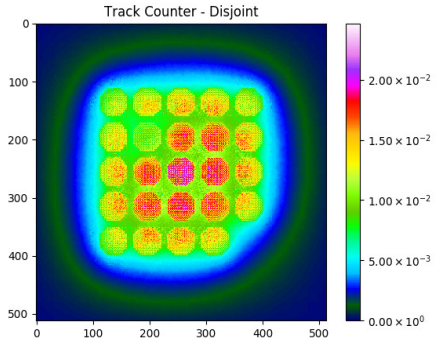


Figure C.262: Track Counter Tally using disjoint tallies. $r_s = 0.75$

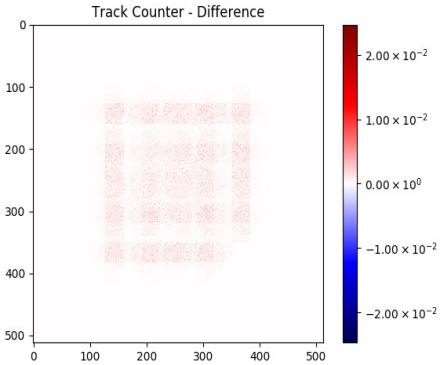


Figure C.263: Absolute Difference in Track Counter Tally between Figure C.261 and Figure C.262 (using min/max of tally)

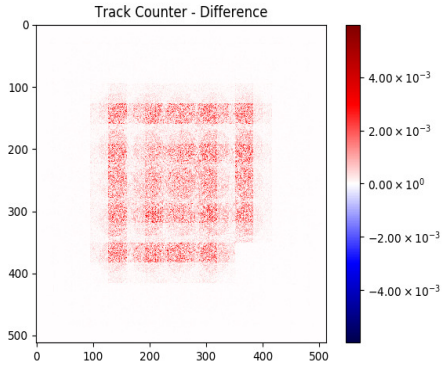


Figure C.264: Absolute Difference in Track Counter Tally between Figure C.261 and Figure C.262 (using min/max of data)

C.2.15 Track Length

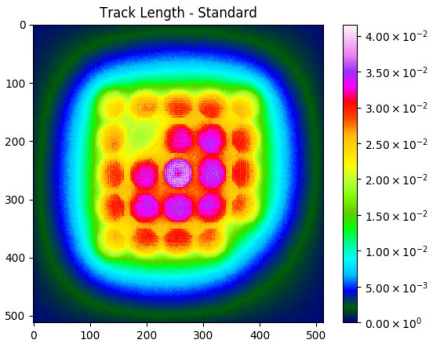


Figure C.265: Track Length Tally using standard tallies

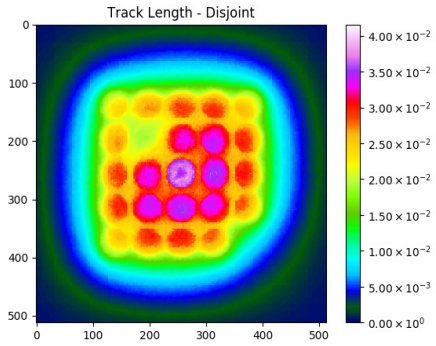


Figure C.266: Track Length Tally using disjoint tallies. $r_s = 0.25$

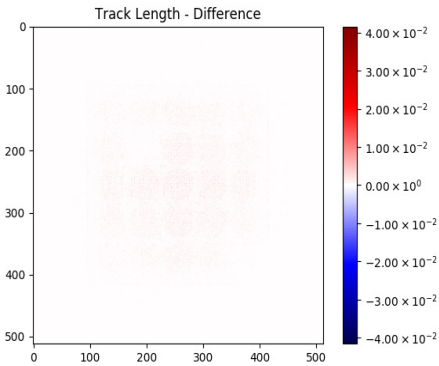


Figure C.267: Absolute Difference in Track Length Tally between Figure C.265 and Figure C.266 (using min/max of tally)

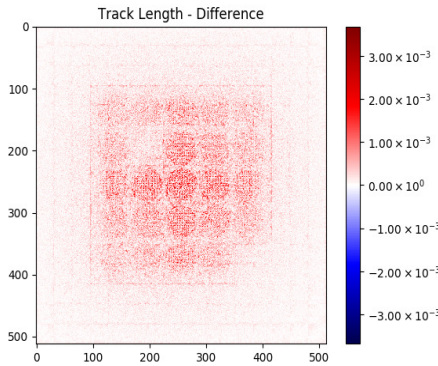


Figure C.268: Absolute Difference in Track Length Tally between Figure C.265 and Figure C.266 (using min/max of data)

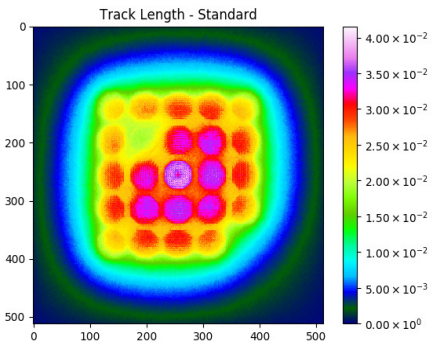


Figure C.269: Track Length Tally using standard tallies

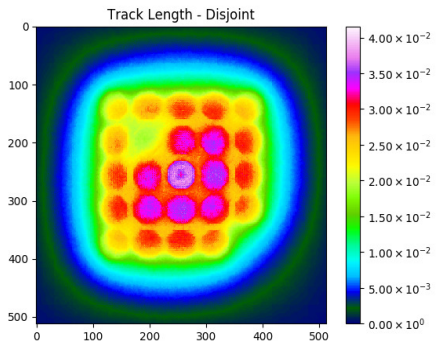


Figure C.270: Track Length Tally using disjoint tallies. $r_s = 0.50$

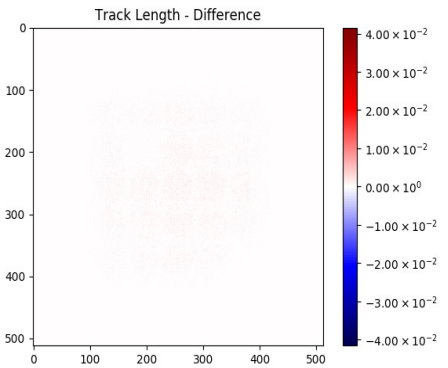


Figure C.271: Absolute Difference in Track Length Tally between Figure C.269 and Figure C.270 (using min/max of tally)

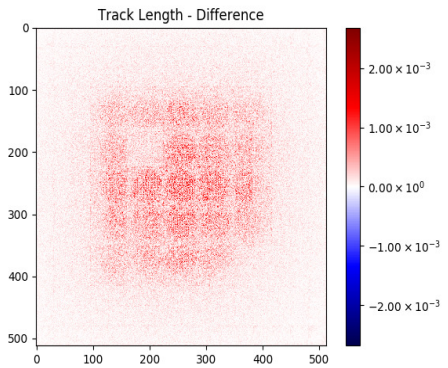


Figure C.272: Absolute Difference in Track Length Tally between Figure C.269 and Figure C.270 (using min/max of data)

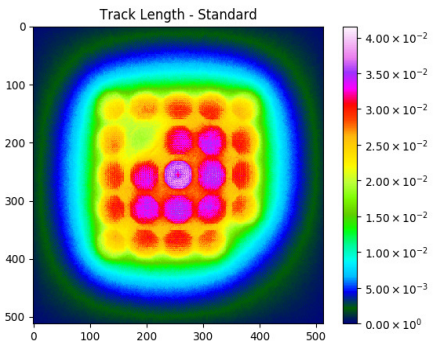


Figure C.273: Track Length Tally using standard tallies

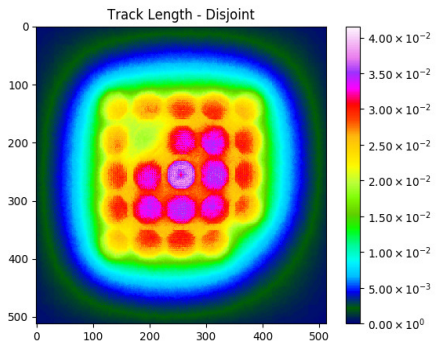


Figure C.274: Track Length Tally using disjoint tallies. $r_s = 0.75$

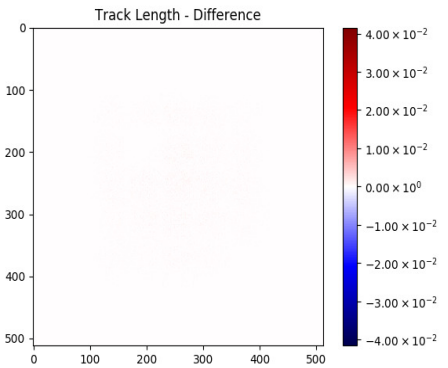


Figure C.275: Absolute Difference in Track Length Tally between Figure C.273 and Figure C.274 (using min/max of tally)

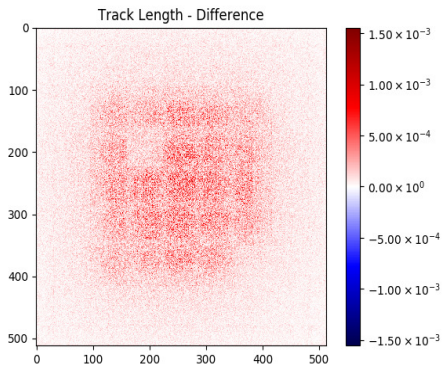


Figure C.276: Absolute Difference in Track Length Tally between Figure C.273 and Figure C.274 (using min/max of data)